EasyPlay User's Guide

October 21, 2011

Copyright © 2011 by HOLTEK SEMICONDUCTOR INC. All rights reserved. Printed in Taiwan. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form by any means, electronic, mechanical photocopying, recording, or otherwise without the prior written permission of HOLTEK SEMICONDUCTOR INC.

Contents

Contents **2**

Chapter 1 Outline **4**

EasyPlay Development Tool Introduction 4

Chapter 2 Quick Start **5**

Starting the EasyPlay 5
Creating a New Project 5
Download and Demo 8

Chapter 3 Project Design **10**

Scenes Dialogues 11
Scripts Dialogues 11
API Dialogues 12
Sound Files Dialogues 12
The Add Button is Used to Add the Sound Files 12
The Del Button is Used to Delete a Sound File 13
Small Technique 13
RAM Dialogues 14
Trigger Dialogues 15
EasyPlay Window Menu 16
Project Menu 16
Utilities Menu 17
Help Menu 17

Chapter 4 Project Mode **18**

Demo Board Hardware Structure 18
HT83F02 Hardware Structure 18
HT83F22 Hardware Structure 18
Mode Function Description 19
Binary Timing 22
I^2C Timing 22
COMBO Timing 23
Holtek Serial Bus Timing 23

Chapter 5 Script API **24**

Assign, Operation 24
Assign 24
Operation 24
VolumeSet, VolumeGet 24
VolumeSet 24
VolumeGet 24
VoicePlay 24
CALL, RET 24
CALL 24
RET 24
Sleep, Stop, Delay_ms, Comment 25
Sleep 25
Stop 25
Delay_ms 25
Comment 25

Jmp_EQU, JumpTo 25
Jmp_EQU 25
JumpTo 25
Check_sum, Triangle_Wave 25
Communication_ISR, KeyTrigger 26
Communication_ISR 26
KeyTrigger 26
Dispatch, DispatchWave 26
Dispatch 26
DispatchWave 27
Input and Output 27
Input 27
Output 27
OutputSharePort 28
Record 28
RecordPlay 28
RecordErase 28
RecordEraseAll 28
ReadRecordNum 28
UserAPI0/1/2/3 28

Chapter 6 Application Examples 29

Key Applications 29
Function Requirements 29
Functional Description 29
Demonstration Figure Description 31
EasyPlay Trigger Window Content 37
Programming and Demo Environments 42
Key and Transmission Applications 44
Key Requirements 44
Recording and Audio Function Applications 47
Function Requirements 47
Functional Description 48
Programming and Demo Environments 49

Chapter 1 Outline

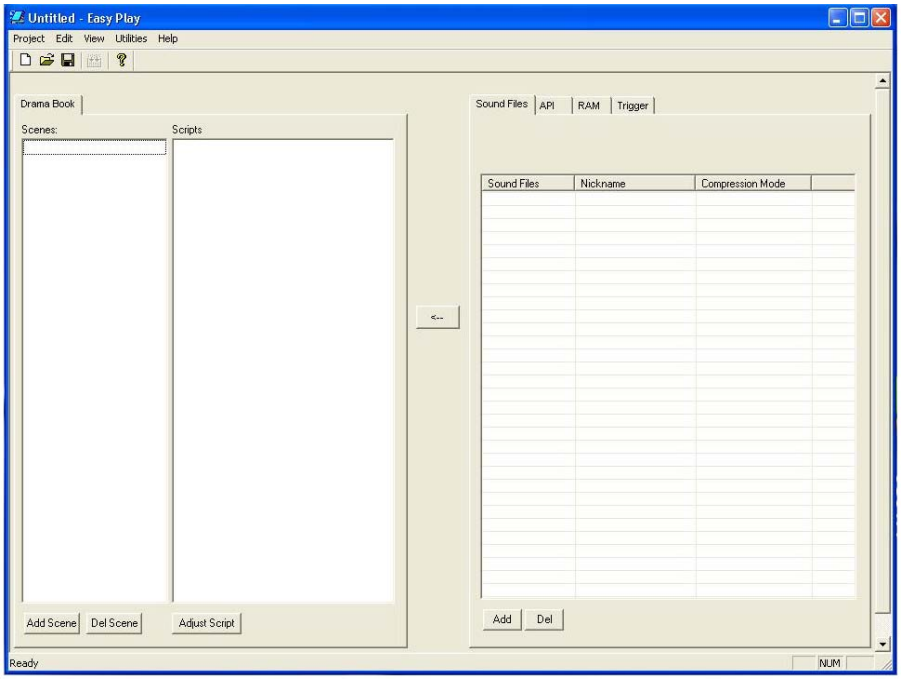
EasyPlay Development Tool Introduction

The EasyPlay Development Tool is an easy to use software development tool designed for the HT83Fxx series of voice devices. It is an interface development tool that can simulate in a single project items such as scenes, scripts, sound files etc. ensuring that designers have their demo boards and projects up and running as fast as possible. For projects with a large number of voice segments, the EasyPlay development tool also provides a superior and greatly simplified development process. EasyPlay stores scenes, scripts and sound files with a certain format in the VROM of the HT83Fxx. As the VROM in the HT83Fxx device series is Flash type memory, the implementation of voice and functional updates becomes a much simplified and convenient process, and one which does not require HT83Fxx device replacement. An example could be any kind of voice function electrical appliance, such as an electronic scale, which although perhaps initially using English, could be very easily and quickly converted to use other languages.

Chapter 2 Quick Start

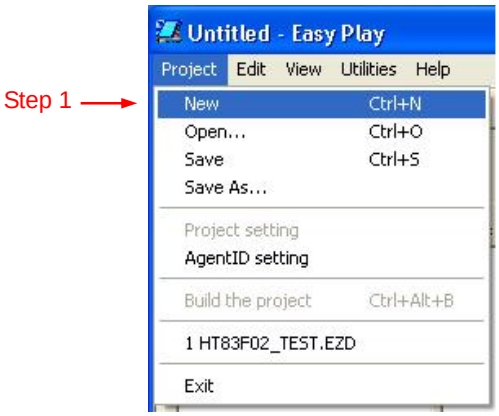
Starting the EasyPlay

Double click the EasyPlay icon

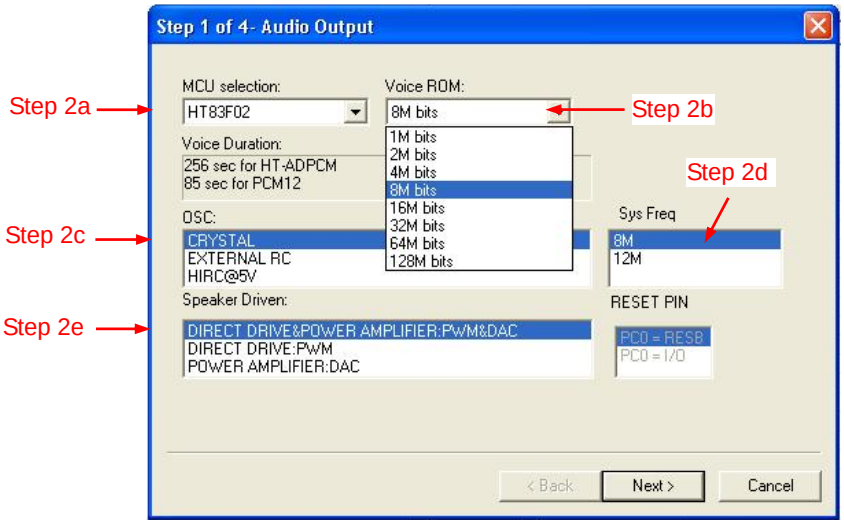


Creating a New Project

Step1. Choose "New"



Step2. Audio Output Window



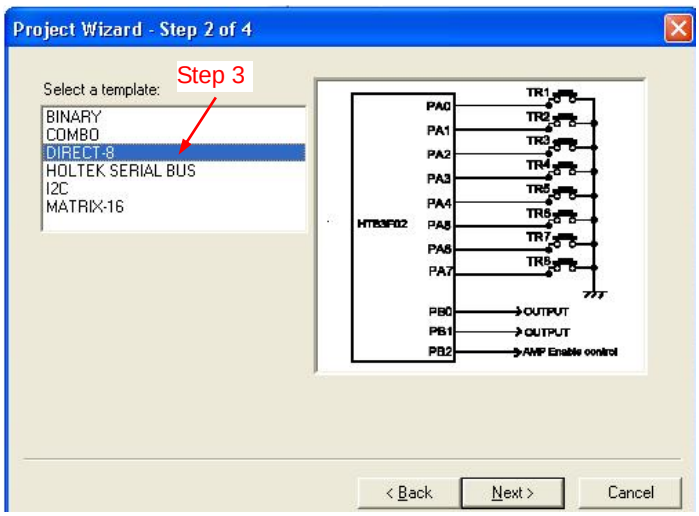
There are 5 configuration options:

- Step 2a. MCU Body
- Step 2b. Voice ROM Capacity
- Step 2c. MCU Oscillator Selection
- Step 2d. MCU Operating Frequency
- Step 2e. Speaker Drive Selection

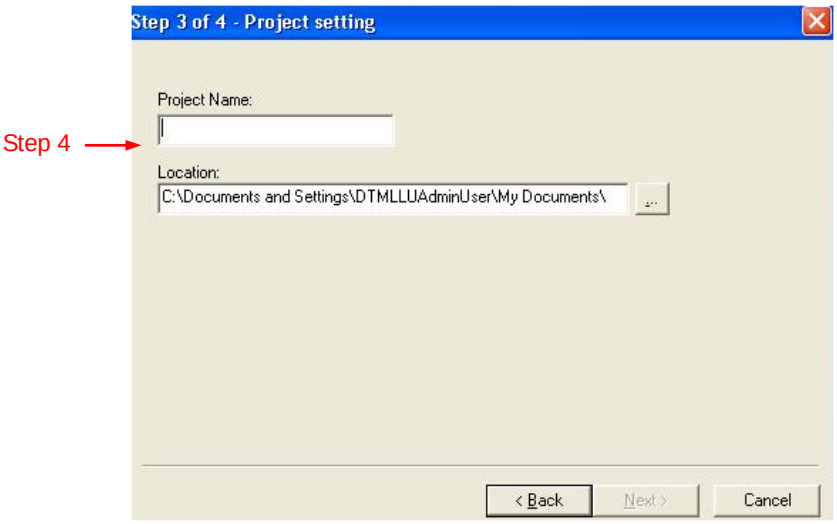
After selecting the required configuration options, click the "Next" button.

Step3. There are six operating modes within the EasyPlay.

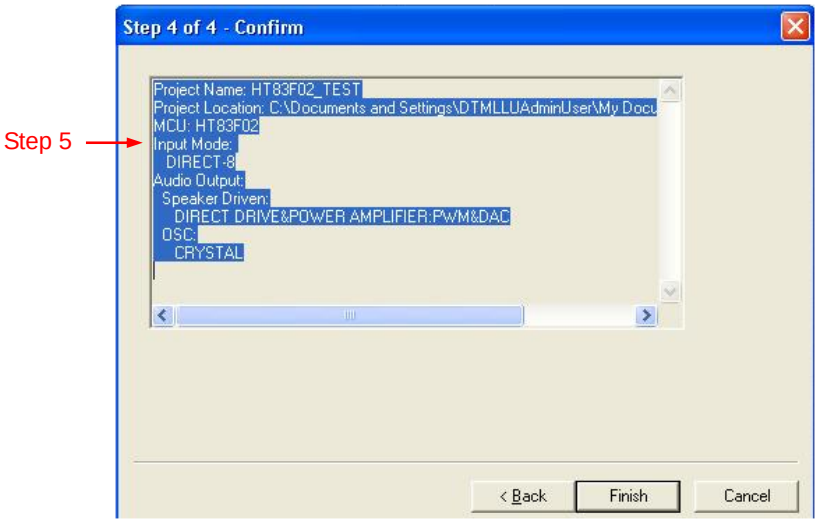
Choose DIRECT-8 for this example. The other operating modes will be introduced later.



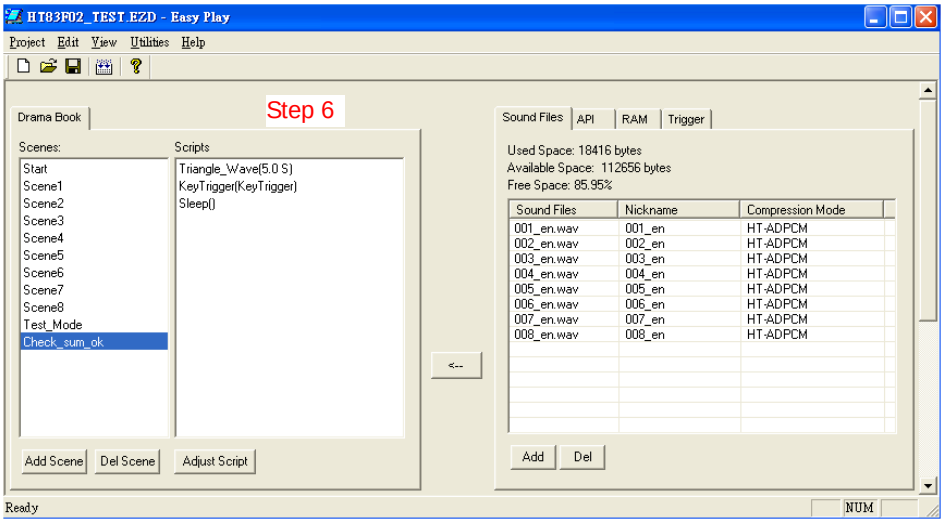
Step4. Input the project name and setup the storage location.



Step5. Confirm all settings

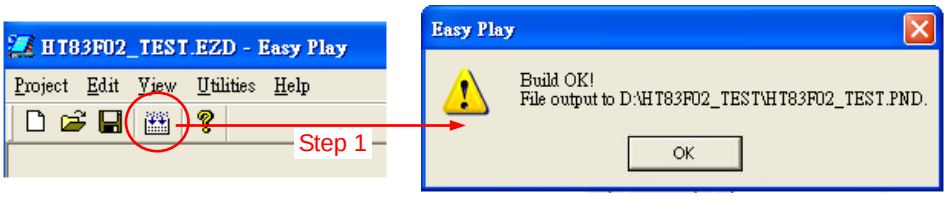


Step6. New Project Now Created

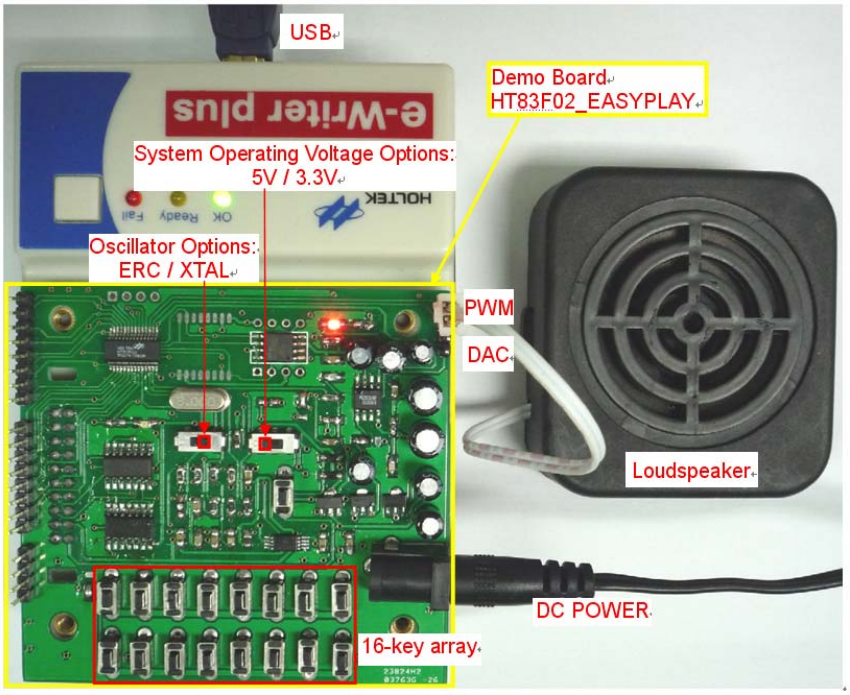


Download and Demo

Step1. Generate a PND File



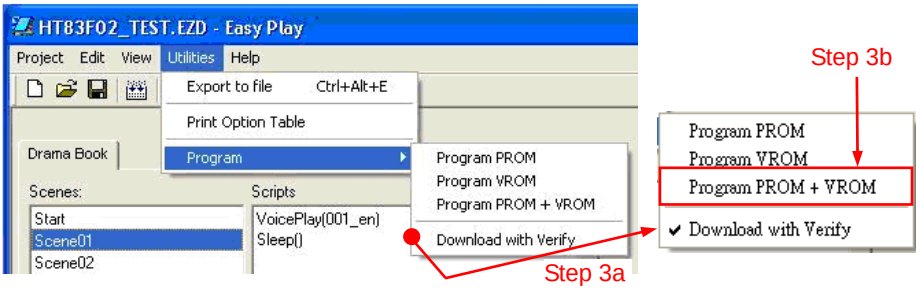
Step2. Setup the Hardware Platform



Step3. Select the programming method according to requirements. The other options will be introduced later.

Step3a. First choose the calibration method for the programming process and for pre-programming.

Step3b. Click "Program PROM +VROM" to start programming the program data and voice data.



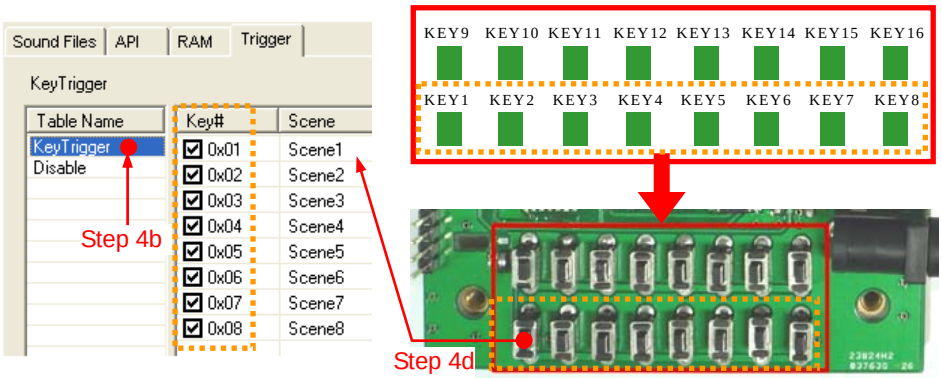
Step4. Demo Method

Step 4a. After programming, then use the hardware as described in Step 2

Step 4b. In the Trigger dialogue box, select "KeyTrigger"

Step 4c. Corresponding key sketch map - Demo Board HT83F02_EASYPLAY:

A. Software Platform Key Numbers (Key#) B. Hardware Platform Key Number

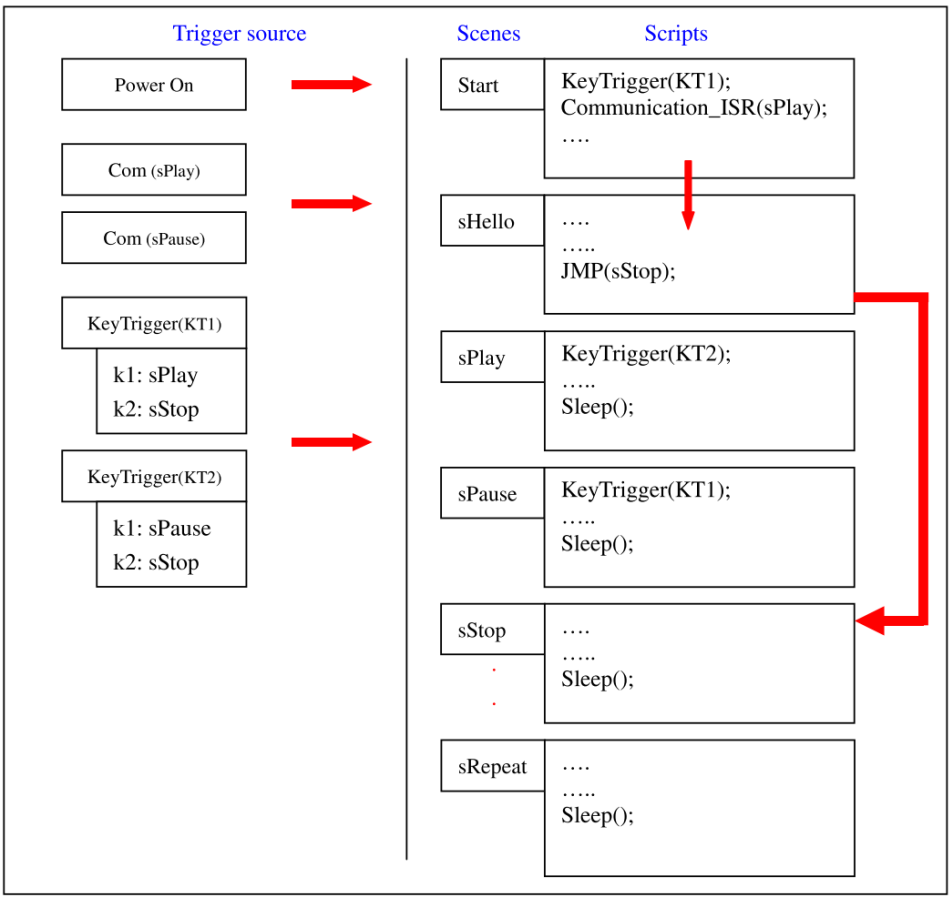


Step 4d. Click the button to enable the required function (play the numbered voice)

Chapter 3 Project Design

The EasyPlay provides Scenes, Scripts, API, Sound Files, RAM and Trigger dialogues. A complete project is divided into several Scenes, each of which corresponds to one Script composed of API, Sound Files, RAM and Trigger. The relationship of each scene, except for the one located in the first location, is a parallel one. If the script related to the scene does not have an API for scene jumping, the scene and its successive scenes are will have a consecutive relationship. The APIs in the script dialogue have a consecutive relationship. Therefore, the first step of a project design is to add scenes to the Scenes dialogue and then have scenes correspond to the script content according to the project requirements. Each scene connection is implemented by a scene jump API.

Note: A hidden scene jump is generated by an external key trigger or a communication trigger. Therefore, if a key trigger or communication trigger is configured during the scene initialisation, when either of them occurs, the MCU will jump to the configured scenes. The operation process is as follows.

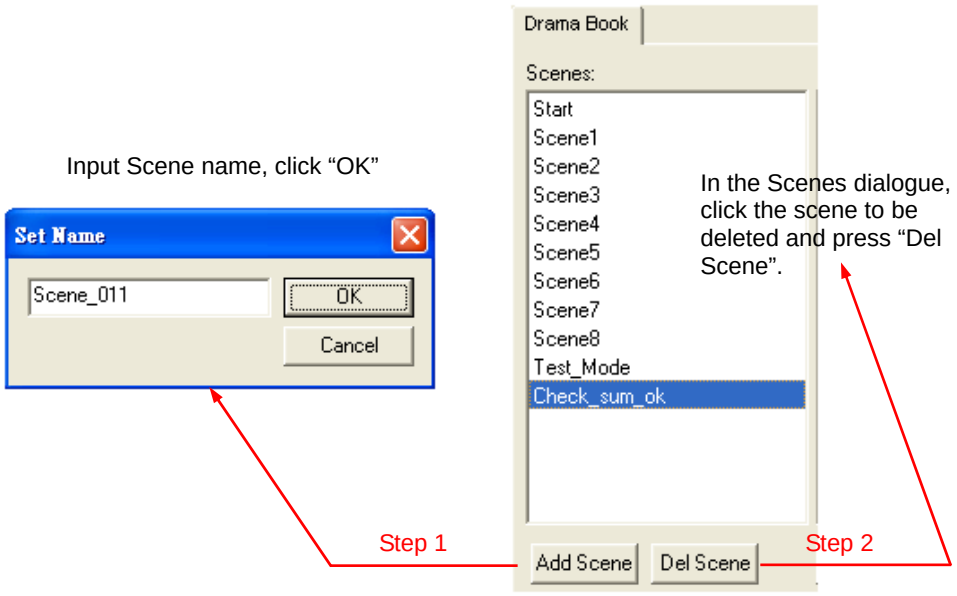


Scenes Dialogues

Scenes dialogue includes Add Scene and Del Scene operating buttons.

Step 1. The Add Scene button is used to add a scene in the Scenes dialogue.

Step 2. The Del Scene button is used to delete a scene in the Scene dialogue.



Note: The first scene is executed when the MCU receives a power on reset. The scene is placed at the first scene location in the Scenes dialogue.

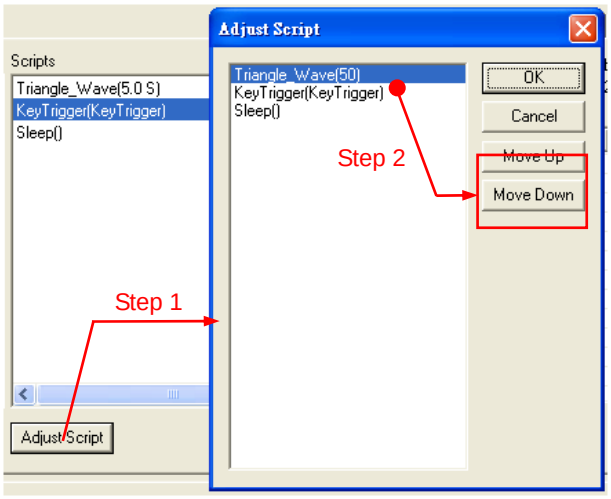
Scripts Dialogues

The Scripts dialogue includes Adjust Script, Add <-- and Delete operating buttons.

(A) The Adjust Script button is used to adjust the scripts sequence.

Step 1. Click the Adjust Script button to pop up the Scripts dialogue.

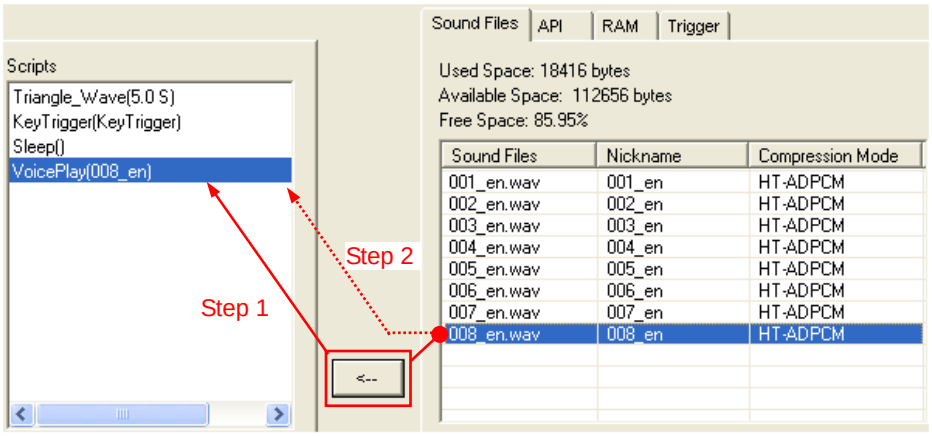
Step 2. Click the script to be adjusted, click the Move Up or Move Down button to shift the script up or down.



Add <-- Button

Step 1. Select the required script from the dialogue box on the right, and click the <-- button.

Step 2. A script can also be added by double clicking the required script file.



(C) Delete a script: Select the script to be deleted and press the "DEL" button.

API Dialogues

The API dialogue includes the API required by the Scripts. By using add <-- button, the API will be added to the Scripts.

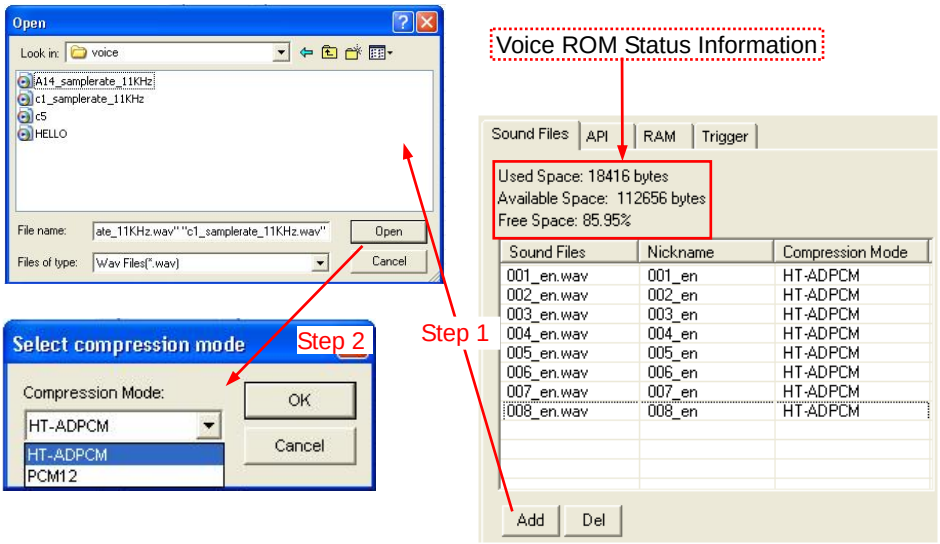
Sound Files Dialogues

The Sound Files dialogue includes Add and Del operating buttons.

The Add Button is Used to Add the Sound Files

Step 1. Click "Add" to pop up the window and find the .WAV file to be added.

Step 2. Click "Open" to pop up the format selection window.



Note: (1) The WAV file must be in a MONO WINDOWS PCM format with a sound frequency sampling of about 16bits.

(2) The relationship between the maximum sampling rate and compression format of the sound file (.wav) is listed as the table below for an 8MHz system frequency.

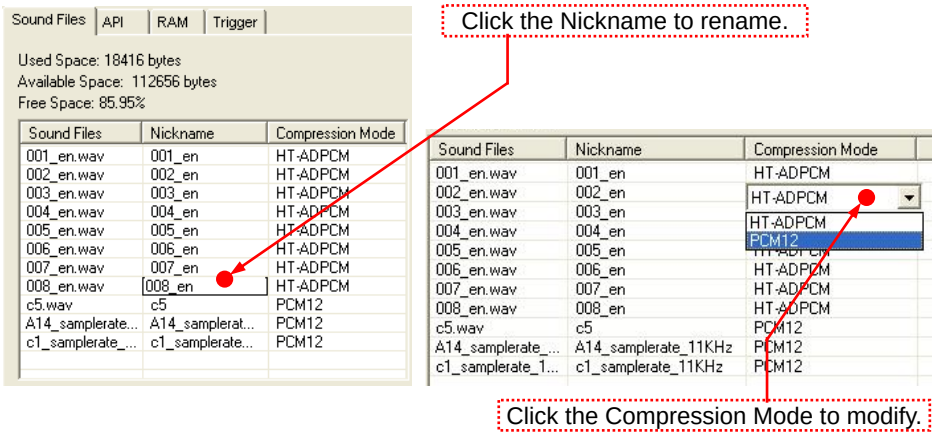
Compression Format	Maximum Sound Sampling Rate
HT-ADPCM	15K
PCM12	18K

The Del Button is Used to Delete a Sound File

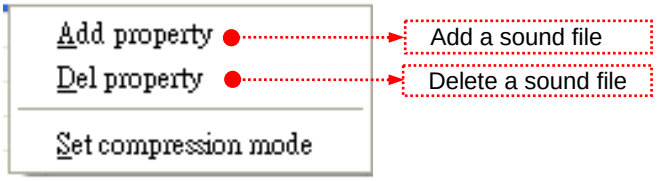
Click the Del button to delete the sound file to be deleted from the Sound Files page.

Small Technique

(1) Choose the sound file from the Sound Files page.

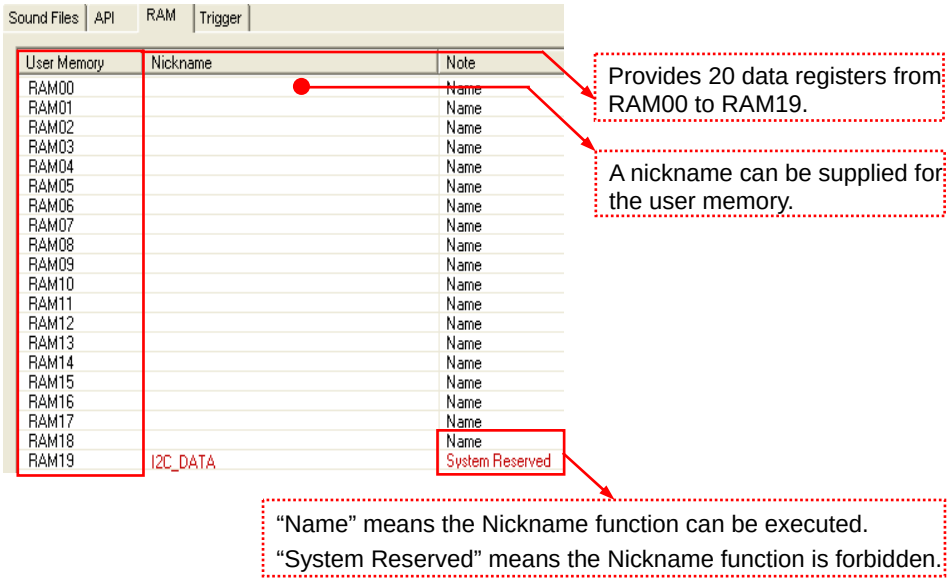


(2) Click the right key after a sound file is chosen to pop up the modification menu.



RAM Dialogues

Click the RAM button and then the EasyPlay interface will change as shown:



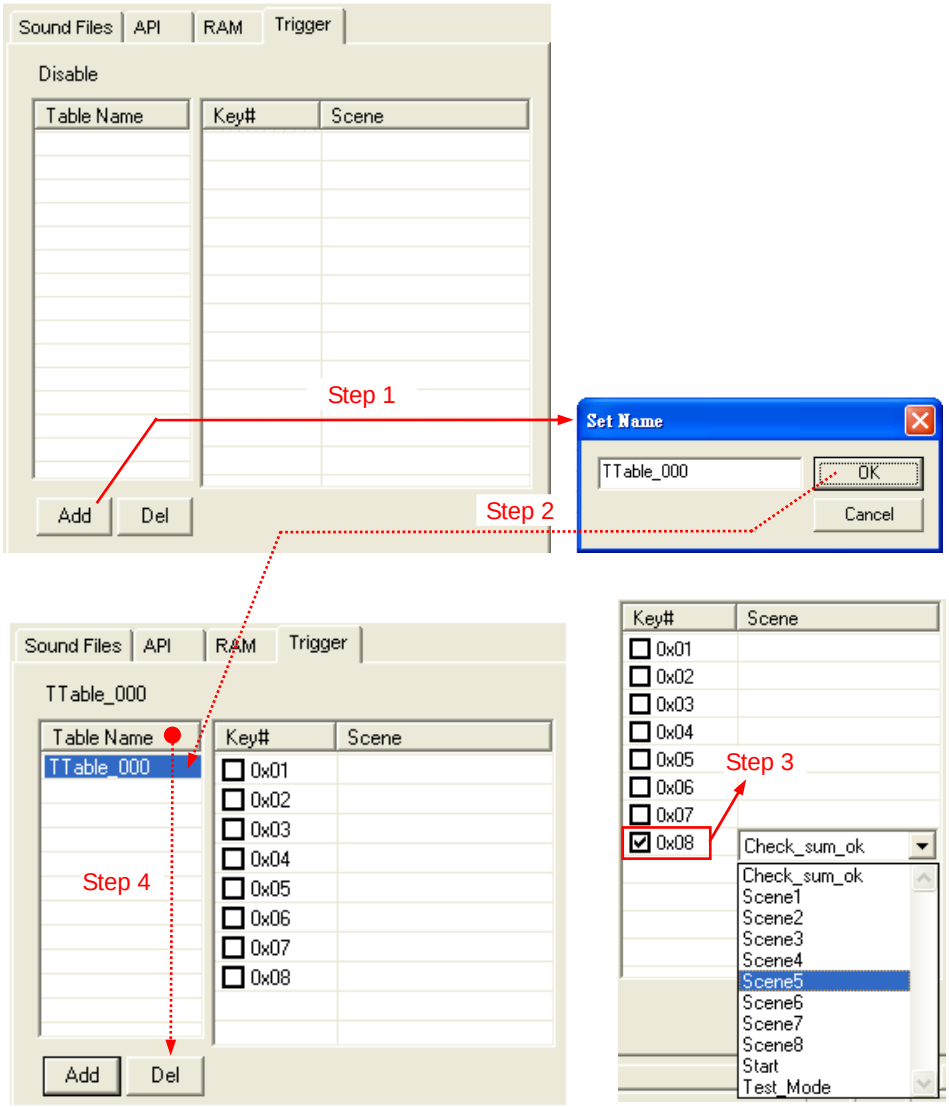
Note: (1) When the Check_sum() API is used, RAM18 is used to store the calibration and result.

(2) For Binary, I^2C, Holtek bus and Combo modes, RAM19 is used to store the received data byte.

Trigger Dialogues

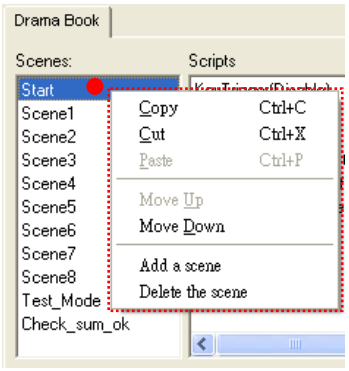
Click the Trigger button to change the EasyPlay to the following:

- The Trigger dialogue is used to display the function configuration of the input key, which is only activated when the "Direct-8", "Matrix-16" or "COMBO" modes is selected.
- The Trigger dialogue includes Add and Del buttons.
  - Step 1. Click "Add" to pop up the Set Name window.
  - Step 2. Display all function configurations of the 8 input keys.
  - Step 3. Click the Key# box to setup the function scene of the key.
  - Step 4. The Del button is used to delete the key function table in the Trigger dialogue.

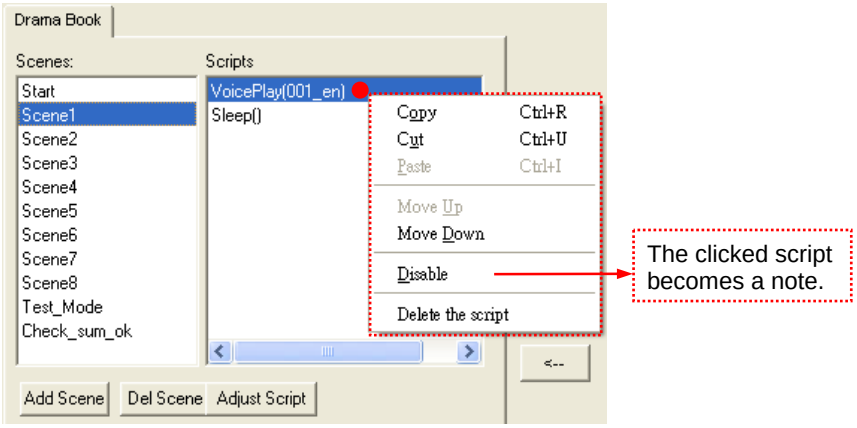


Small Technique:

- In the Scenes dialogue, click the right key to pop up a function menu.



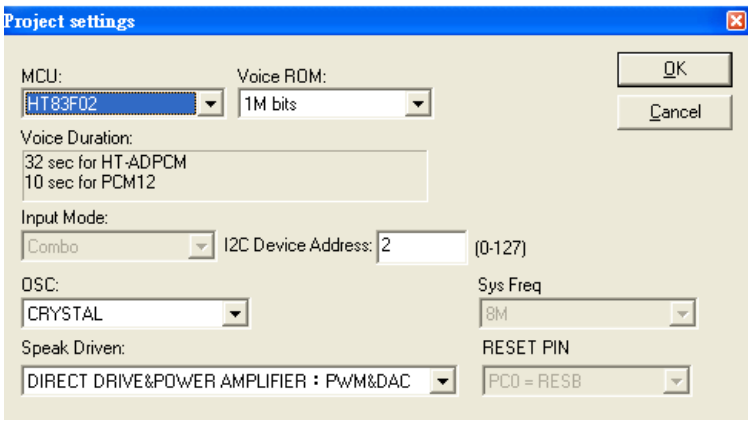
- In the Scripts dialogue, click the right key to pop a function menu.



EasyPlay Window Menu

Project Menu

(1) Project/Project setting: Modifies the project setting.

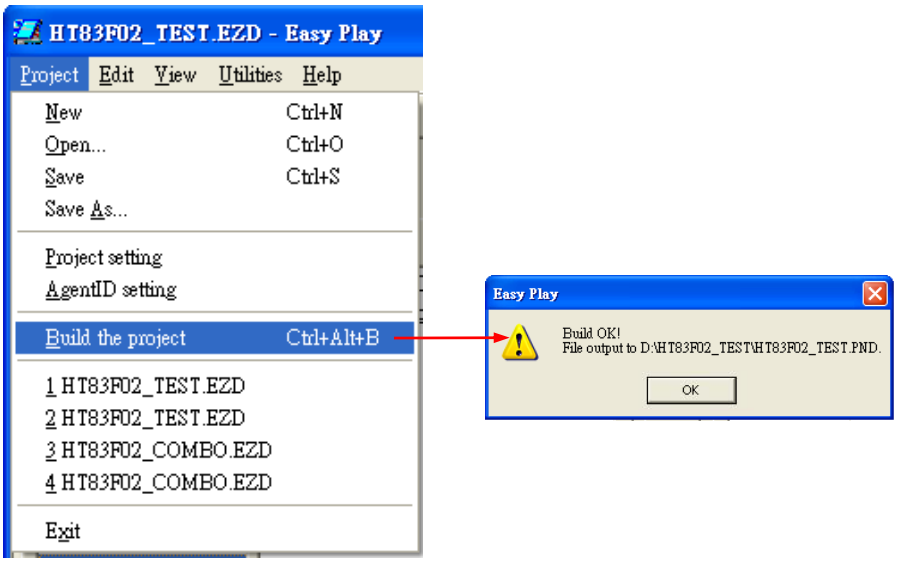


Note: The "MCU" modification can be executed only within bodies of the same type.

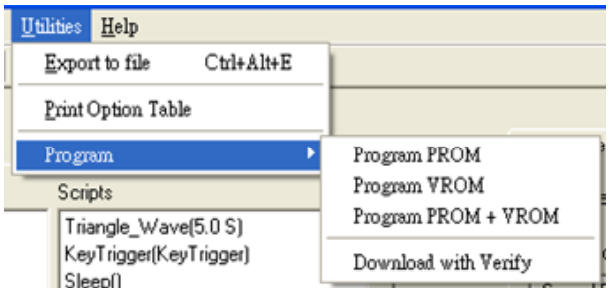
Group A: HT83F02-1Mb/2Mb/4Mb/8Mb/16Mb/32Mb/64Mb/128Mb

Group B: HT83F22-1Mb/2Mb/4Mb/8Mb/16Mb/32Mb/64Mb/128Mb

(2) Project/Build the project: Creates a .pnd project file.



Utilities Menu



- Export to file: Exports the Scenes and Scripts content in the EasyPlay as a text file.
- Print option table: Prints the option info etc. as an Approved Sheet text file.
- Program: Programs the .PND file into the MCU and the voice Flash with four options.
  - Program PROM: Programs the program code only.
  - Program VROM: Programs the voice data code.
  - Program PROM+VROM: Programs both the program code and voice data code.
  - Download with Verify: Calibrates the program process.

Note: For Flash type devices, after compilation, the Project/Project setup must select the Program PROM+VROM for programming so as to implement the correct modification.

Help Menu

The Help menu includes information such as the EasyPlay version info, copyright date option and the EasyPlay user's menu option.

Chapter 4 Project Mode

Demo Board Hardware Structure

HT83F02 Hardware Structure

Name	PA0	PA1	PA2	PA3	PA4	PA5	PA6	PA7	PB0	PB1	PB2	PC0~1	PC2
Direct-8	K0	K1	K2	K3	K4	K5	K6	K7	O0	O1	/BZ	SO0-SO1	SO2
Matrix-16	R0	R1	R2	R3	R4	R5	R6	R7	C1	C2	/BZ	SO0-SO1	SO2
Binary	D0	D1	D2	D3	D4	D5	D6	K7	O0	O1	/BZ	SO0-SO1	SO2
I^2C	O0	O1	O2	O3	O4	O5	O6	O7	SDA	SCL	/BZ	SO0-SO1	SO2
Holtek	O0	O1	O2	O3	O4	O5	DATA	CLK			/BZ	SO0-SO1	SO2
Combo	K0	K1	K2	K3	K4	K5	K6	K7	SDA	SCL	/BZ	SO0-SO1	SO2

Note: K0~K7 = input keys
O = output
R = input
C = column output
/BZ = power control signal
D = data
SDA/SCL = I2C data and clock lines
DATA/CLK = Holtek communication data and clock lines
SO = common ports (PC0-PC2), which are used as output ports when used as I/O.

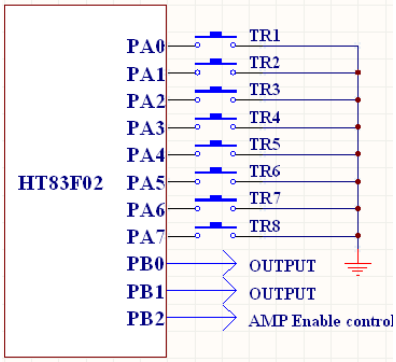
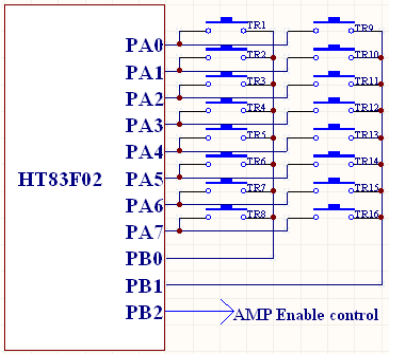
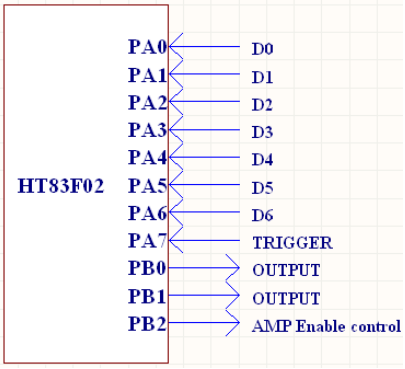
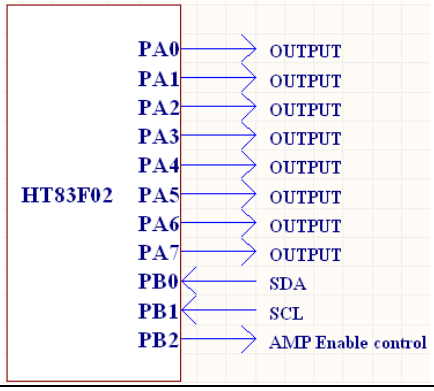
HT83F22 Hardware Structure

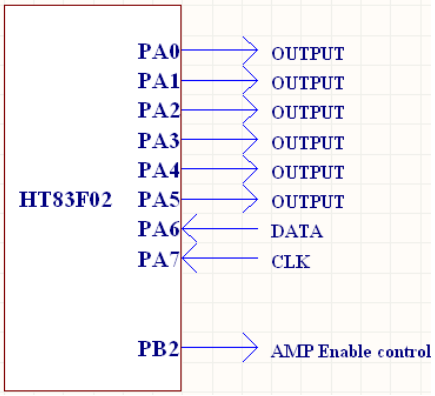
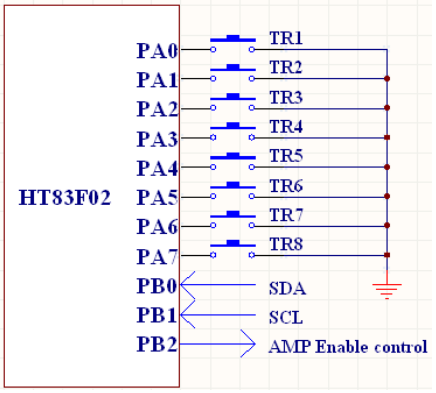
Name	PA2	PA3	PA4	PA5	PA6	PA7	PA1	PB5	PB6	PB7	PB0~4	PC0~7
Direct-6	K1	K2	K3	K4	K5	K6	ES	FL	RD	/BZ	SO	O0~O7
Matrix-8	R0	R1	R2	R3	C0	C1	ES	FL	RD	/BZ	SO	
Binary	D0	D1	D2	D3	D4	K6	ES	FL	RD	/BZ	SO	
I^2C	I0	I1	I2	I3	SDA	SCL	ES	FL	RD	/BZ	SO	
Holtek	I0	I1	I2	I3	DATA	CLK	ES	FL	RD	/BZ	SO	
Combo	K0	K1	K2	K3	SDA	SCL	ES	FL	RD	/BZ	SO	

Note: K1-K6 = input keys
O = output
I = input
R = matrix row key input
C = matrix column key input
/BZ = the HT82V739 enabled signal
D = data
SDA/SCL = I2C data and clock lines
DATA/CLK = Holtek communication data and clock lines
RD = "busy" recording signal
ES = record erase indicator
FL = write SPI FLASH or erase SPI FLASH failed indicator
SO = selective output, namely when PB0,PB1 and PB2 are selected to be I/O ports, then PB0,PB1 and PB2 will they be used as output ports and PB3-4 will be used as fixed output ports.

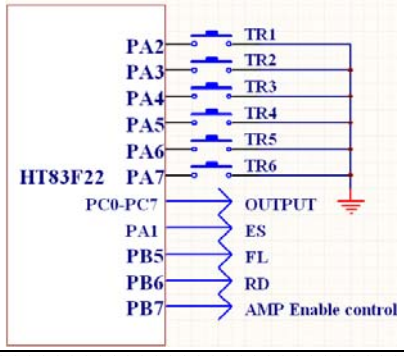
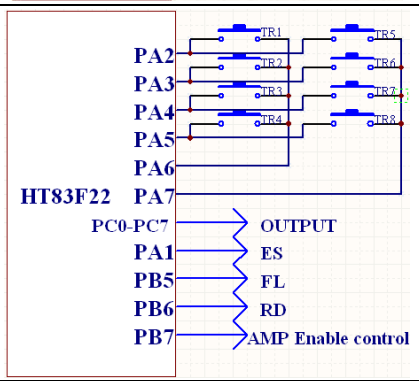
Mode Function Description

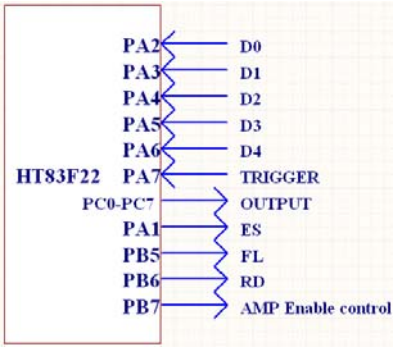
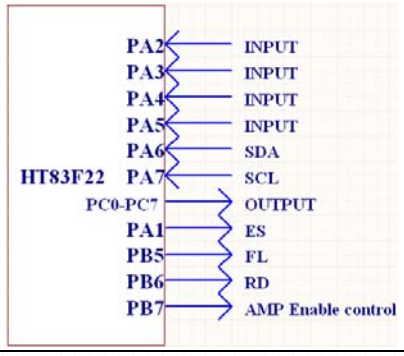
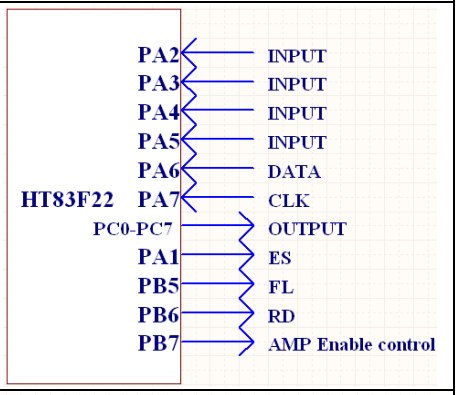
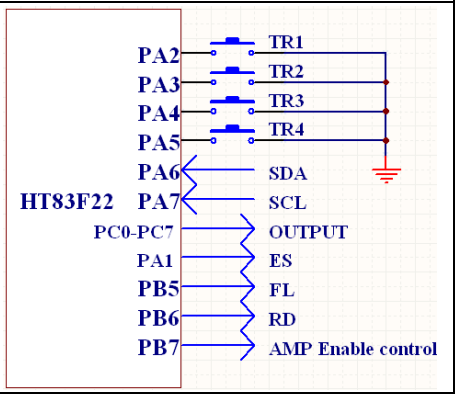
Taking the HT83F02 as an example, the description is as follows:

Mode	Function Description	HT83F02 I/O Input/Output Sketch Map
Direct-8	1. Uses PA as an output port to connect the 8 keys to ground 2. Uses PB0~PB1 as pins for user output ports 3. Uses PB2 as the power control output port	
Matrix-16	1. Uses PA, PB1 and PB2 ports to form an 8x2 matrix keyboard with PA as inputs, PB0 and PB1 as outputs. 2. Uses PB2 as the power control output port	
Binary	1. Uses PA0~PA6 and PA7 ports to form parallel trigger inputs 2. Uses PA7 as the trigger input and PA0~PA6 as parallel output ports. 3. See "Binary Timing" for its timing reference. The data is saved in RAM19.	
I^2C	1. Uses PB0~PB1 as I^2C Slave ports 2. Uses PA0~PA7 as user output ports 3. Uses PB2 as the power control output port 4. See "I^2CTiming" for the timing reference. The data is saved in RAM19.	

Mode	Function Description	HT83F02 I/O Input/Output Sketch Map
Holtek	1. Uses PA6~PA7 as Holtek serial bus Slave ports 2. Uses PA0~PA5 as user output ports 3. Uses PB2 as the power control output port 4. See "Holtek Serial Bus Timing" for the timing reference. The data is saved in RAM19.	
Combo	1. Uses PB0~PB1 as the I^2C Slave ports 2. Uses PA as an input port to connect the eight keys to ground 3. Uses PB2 as the power control port 4. See "COMBO Timing" for the timing reference. The data is saved in RAM19.	

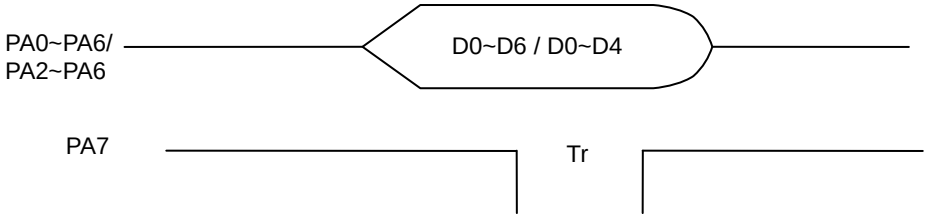
Taking the HT83F22 as an example, the description is as follows:

Mode	Function Description	HT83F22 I/O Input/Output Sketch Map
Direct-6	1. PA2~PA7 ports connect the six keys to ground 2. Uses PA1 as an SPI Flash Erase indicator 3. Uses PC0~PC7 as user ports 4. Uses PB5 as a write or erase SPI Flash indicator 5. Uses PB7 as the power control output	
Matrix-8	1. Uses PA2~PA5 and PA6~PA7 to form a 4×2 matrix keyboard with PA2~PA5 as column inputs and PA6 and PA7 as row inputs 2. Uses PA1 as a SPI Flash Erase indicator 3. Uses PC0~PC7 as user output ports 4. Uses PB5 as a write or erase SPI Flash indicator 5. Uses PB7 as the power control output port	

Mode	Function Description	HT83F22 I/O Input/Output Sketch Map
Binary	1. Uses PA2~PA6 and PA7 to form a column trigger input 2. Uses PA7 as a trigger input port with PA2~PA6 as parallel data input ports 3. Uses PA1 as an SPI Flash Erase indicator 4. Uses PC0~PC7 as user output ports 5. Uses PB5 as a write or erase SPI Flash indicator 6. Uses PB7 as the power control output port 7. See "Binary Timing" for the timing reference. The data is saved in RAM19.	
I^2C	1. Uses PA6~PA7 as the I^2C Slave port 2. Uses PA2~PA5 as user outputs 3. Uses PA1 as an SPI Flash Erase indicator 4. Uses PC0~PC7 as user output ports 5. Uses PB5 as a write or erase SPI Flash indicator 6. Uses PB7 as the power control port 7. See "I^2C Timing" for timing reference. The data is saved in RAM19.	
Holtek	1. Uses PA6~PA7 as the Holtek serial bus Slave port 2. Uses PA2~PA5 as user output ports 3. Uses PA1 as a SPI Flash Erase indicator 4. Uses PC0~PC7 as user output ports 5. Uses PB5 as a write or erase SPI Flash indicator 6. Uses PB7 as the power control output port 7. See "Holtek Serial Bus Timing" for timing reference. The data is saved in RAM19.	
Combo	1. Uses PA6~PA7 as the I^2C Slave port 2. Uses PA2~PA5 to connect the four keys to ground 3. Uses PA1 as an SPI Flash Erase indicator 4. Uses PC0~PC7 as user output ports 5. Uses PB5 as a write or erase SPI Flash indicator 6. Uses PB7 as the power control port 7. See "COMBO Timing" for the timing reference. The data is saved in RAM19.	

Binary Timing

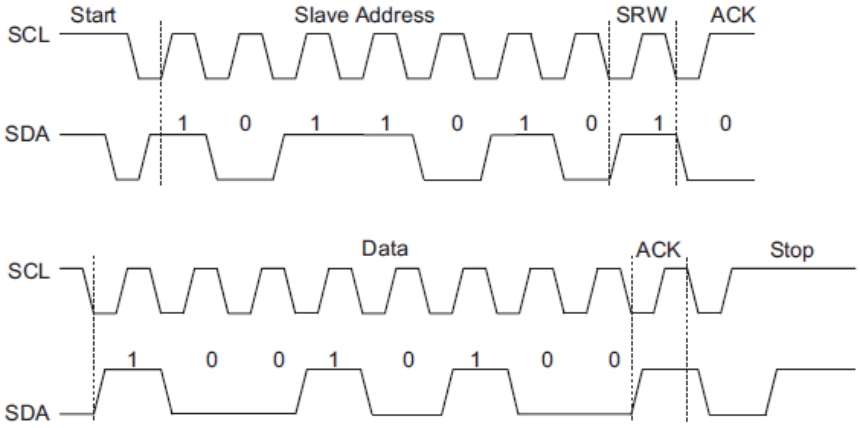
- Timing Figure



- The input trigger time (Tr) of PA7 is 10~20ms.
- The time interval between two successive communications must be less than 150us.
- Each communication data byte is saved in RAM19.

I²C Timing

- For I²C communication, the Master timing must obey the following timing for the Slave, namely the HT83Fxx, to response correctly.



S=Start (1 bit)
SA=Slave Address (7 bits)
SR=SRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (RXAK bit for transmitter, TXAK bit for receiver 1 bit)
P=Stop (1 bit)

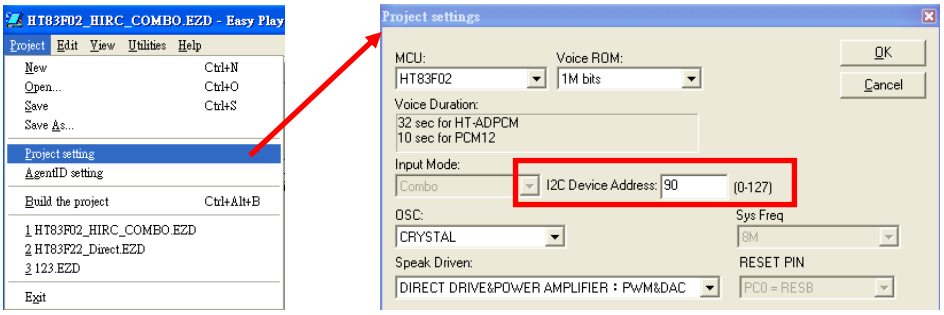
S	SA	SR	M	D	A	D	A	--------	S	SA	SR	M	D	A	D	A	--------	P

I²C Communication Timing Diagram

- Each I²C communication is composed of the following content: S, SA, SR, M, D, A and P. The interval of two successive communications should not be less than 150us.
- Each data byte is saved in RAM19.

Note: (1) After the HT83Fxx receives a Slave Address (SA) and responds (ACK), SCL will be set to a low level. Therefore the external MCU should wait for the HT83Fxx to release the SCL before data transmission.

(2) In the timing figure above, the Slave Address from the master is B4H (180) while the I²C Device Address is set to 90.

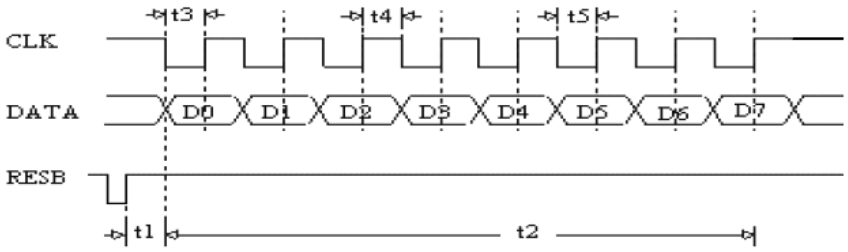


COMBO Timing

- This is a combination mode using both I^2C and Direct-Key modes. See the "I^2C Mode" for the I^2C communication timing.
- Each I^2C communication is composed of S, SA, SR, M, D, A and P. The interval between two successive communications should not be less than 150us.
- Each communication byte data is saved in RAM19.

Holtek Serial Bus Timing

- For Holtek serial bus communication, the Master timing must obey the following timing order for the Slave HT83Fxx device to response correctly.



- t1: The time for the MCU to receive instructions after power up. t1 must be greater than 400ms.
- t2: The time for the MCU to receive 8-bits of data, in other words 1 byte. This will be invalid if the MCU is unable to receive one complete data byte within the time span. To receive another byte, t2 must be less than 50ms.
- t3: The time for the MCU to be woken up to receive an input byte. t3 must be greater than 10ms and less than 18ms.
- t4: The positive half CLK cycle period for the synchronised data. t4 must be greater than 300us and less than 4ms.
- t5: The negative half CLK cycle period for the synchronised data. t5 must be greater than 300us and less than 4ms.
- The interval of two successive communications must not be less than 150us.
- Each communication data byte is saved in RAM19

Chapter 5 Script API

The API is divided into two parts, one part refers to all MCU devices and the other part only refers to a specific MCU device. In the following API description, part one to ten refers to the common API, while part 11 refers to the specific API for the HT83F02/22. Parts 12 to 16 refer to the specific API for the HT83F22.

Assign, Operation

Assign

Assign is used to give value to one specific RAM unit with a decimal constant value range of 0 to 255 or RAM content value.

Operation

Operation is used for specific RAM units or one RAM unit for addition (+), subtraction (-), (AND), (OR) and (XOR) operations. The constant is a decimal digit ranging from 0 to 255.

VolumeSet, VolumeGet

VolumeSet

VolumeSet is used to adjust the volume during broadcasting. The volume levels are subdivided into 0, 1, 2, 3, 4, 5, 6, 7 levels. The level 0 is the mute status while the volume increases in amplitude from 1 to 7 levels with a maximum at level 7.

VolumeGet

VolumeGet is used to supply the current volume level value to a specific RAM unit which is selectable from RAM0 to RAM19.

VoicePlay

VoicePlay is used to play the .wav file using the default sampling rate.

CALL, RET

CALL

CALL is used to call a Scene. It must be used with RET and with only one stack.

RET

RET is used to call back a Scene, no parameter provided.

Sleep, Stop, Delay_ms, Comment

Sleep

Sleep implements the sleep function of the system, no parameter provided.

Stop

Stop is used to stop playing sound and goes on executing next API.

Delay_ms

- Delay_ms implements the delay function with the units in ms.
- The delay period ranges from 0 ms to 65535 ms.

Comment

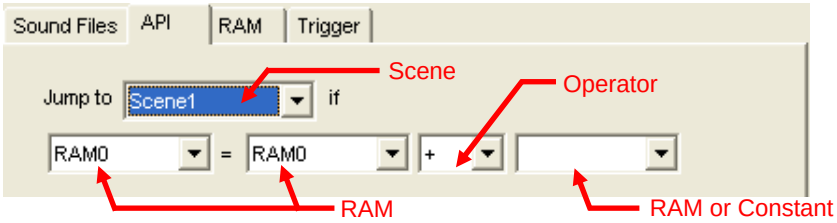
Comment is used to comment Scripts.

Jmp_EQU, JumpTo

Jmp_EQU

Jmp_EQU means conditionally jumping to a specific scene.

- The "Jump to" dialogue is the Scene selection box in which the Scene to jump to can be selected from the menu.
- Dialogue boxes behind "if" is used to setup the "Jump to" conditions. The "Jump to" condition depends on whether the values before and after and equal sign (=) are the same.
- The first and second dialogue boxes in the "Jump to" condition dialogue are the RAM unit selection boxes, ranging from RAM0 to RAM9.
- The third dialogue box is the operator selecting box, including addition (+), subtraction (-), (AND), (OR) and (XOR) operators.
- The forth dialogue is the RAM unit or constant input box with a RAM unit ranging from RAM0 to RAM19 and a constant from 0 to 255.



JumpTo

- JumpTo means is a non conditional jump to a specific scene.
- "Jump to"dialogue is the Scene selection box.

Check_sum, Triangle_Wave

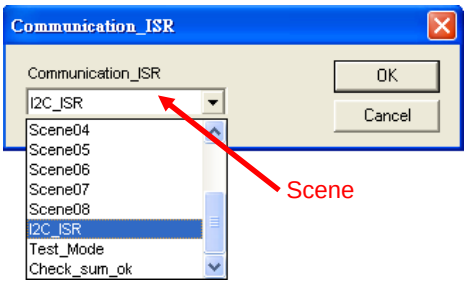
- Check_sum
  Check_sum is used for Voice ROM(SPI) calibration and detection. The result will be saved in RAM18. If RAM18=00H, the calibration is correct or else RAM18=01H. No parameter is provided for Check_sum.
- Triangle_Wave
  Triangle_Wave is used to output a 1K Hz trangle wave from the AUDIO port of the MCU. There is a time parameter for the Triangle_Wave meaning the output time of the triangle wave. The time range is 0 to 25.5s.

Note: If the time parameter is zero, the MCU will continuously output triangle waves.

Communication_ISR, KeyTrigger

Communication_ISR

- Communication_ISR is used to setup a single Scene data communication between the MCU and the external device.



- Communication_ISR can only select Binary, Holtek, Combo or I^2C modes.

KeyTrigger

- KeyTrigger is used is used to setup the external MCU keys, each of which corresponds to a related scene.
- KeyTrigger can only select Direct-8, Matrix-16 and Combo modes.

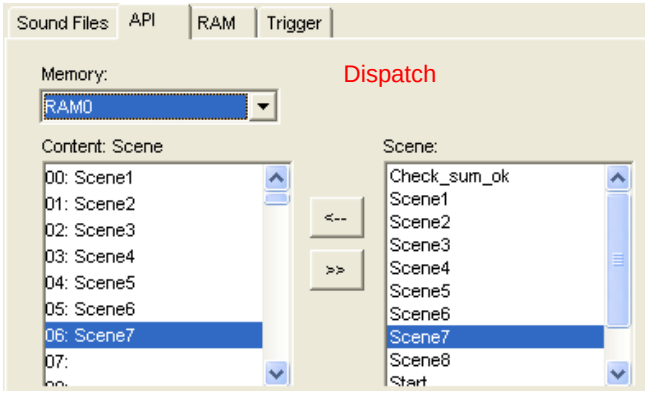
Dispatch, DispatchWave

This is a parallel branching process. When an operation result is generated, it will dispatch the main program to other processing procedures so as to meet the requirement of the application.

Dispatch

- Dispatch is used to transfer data from an indicated RAM value to a scene. When the RAM value is smaller or equal to the number of dispatch scenes, the RAM value corresponds to the specified scene. The dispatch scene order number starts from a zero value.
- When the RAM unit number minus 1 is greater than the dispatch scene, the RAM value corresponds to the first dispatch scene. "Dispatch" includes parameters of the RAM unit and the dispatch scene.
  - The "Memory:" box is the RAM unit selecting menu.
  - The "Scene:" box displays the scenes to be dispatched.
  - The "Content: Scene:" column displays the dispatch scene that has been added. The number of added scenes should be less than or equal to 255.

  The order of the dispatched scenes can be changed by dragging them with the mouse.



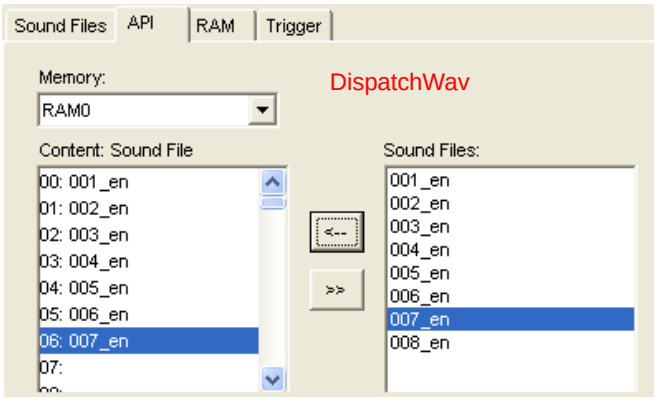
- The <-- button is to add a dispatch scene. To operate first select the location to be added from the "Content: Scene" then the required scene from the "Scene" column by clicking <--.
- The >> button deletes all dispatched scenes. Clicking on >> will remove all of the "Content Scene" content.

Note: (1) It is acceptable for an "empty scene" to be a dispatch scene. However an effective empty scene must be located between or before a non-empty scene.

(2) A jump to an "empty scene" means no jump.

DispatchWave

- The DispatchWave is used to play the .wav file in the DispatchWave according to the specific RAM value. The sequence of the RAM value corresponds with that of the .wav file in the DispatchWave. The order number starts from zero. When the number of RAM values are less than the number of the .wav files added to the DispatchWave, both will corresponds to each other.
- When the RAM value is larger than the number of.wav files added to DispatchWave, the RAM value that is larger than the .wav file number will correspond to the first .wav file in DispatchWave.



Note: (1) The number of the DispatchWave is smaller than or equal to 255.

(2) The "Content: Sound File" in the DispatchWave allows an "empty space", yet an effective empty space should be located between or before the sound files. The "empty space" means no sound will be played.

Input and Output

Input

Input is used to read the MCU PA port status and transfer it to the user RAM. No parameters are available for the Input.

Output

Output is used to transfer the user specified RAM value to the port selected to be an output, such as the PB0-PB2 ports of the HT83Fx0 in the Direct-8 mode.

OutputSharePort

- OutputSharePort is the specified API in the HT83F02/22.
- OutputSharePort is used to output the RAM value or constant specified by the user to an I/O port, such as the PC0-PC2 ports in the HT83F02 or the PB0-PB4 ports in the HT83F22.

Record

- Record is the specified API in the HT83F22.
- Record is to record sound according to the specified RAM value order and using an 8kHz sampling rate. When the RAM value is larger than the existing total number of recordings, the current recording will be located exactly after the previous recording, otherwise recording will be executed according to the RAM number sequence.

RecordPlay

- RecordPlay is the specified API in the HT83F22.
- RecordPlay is used to play a previous recording in the sequence as specified by the indicated RAM value. When the RAM value is larger than the current total recording, the last recording file will be played.

RecordErase

- RecordErase is the specified API in the HT83F22.
- RecordErase is used to delete one recorded file from the sequence related to the specified RAM value. When the specified RAM value is larger than the current total recording, the last recording file will be the one to be deleted.

RecordEraseAll

- RecordEraseAll is the specified API in the HT83F222.
- RecordEraseAll is used to delete all recordings.

ReadRecordNum

- ReadRecordNum is the specified API in the HT83F22.
- ReadRecordNum is to read the total number of the current recording.

UserAPI0/1/2/3

- UserAPI0/1/2/3 is the specified API in the HT83F22.
- In the possible system resource range of the HT83F22, the user can write program code with UserAPI and execute those program codes in the flow designated by the user. The program code must be written in the Source file under the a project folder according to the related location of the UserAPI_HT83F22.asm file.
- The UserAPI is divided into two type. One type is the API (UserAPI0/1) without scene jumping functions and the other is the API (UserAPI2/3) with scene jumping functions.

Chapter 6 Application Examples

Key Applications

Function Requirements

- The example uses the HT83F02 MCU.
- Five Keys:
  - K1 -- Play the current sound file
  - K2 -- Play the previous sound file
  - K3 -- Play the next sound file
  - K4 -- Increase volume and play the current sound file
  - K5 -- Reduce volume and play the current sound file
- Play 8 sound files

Functional Description

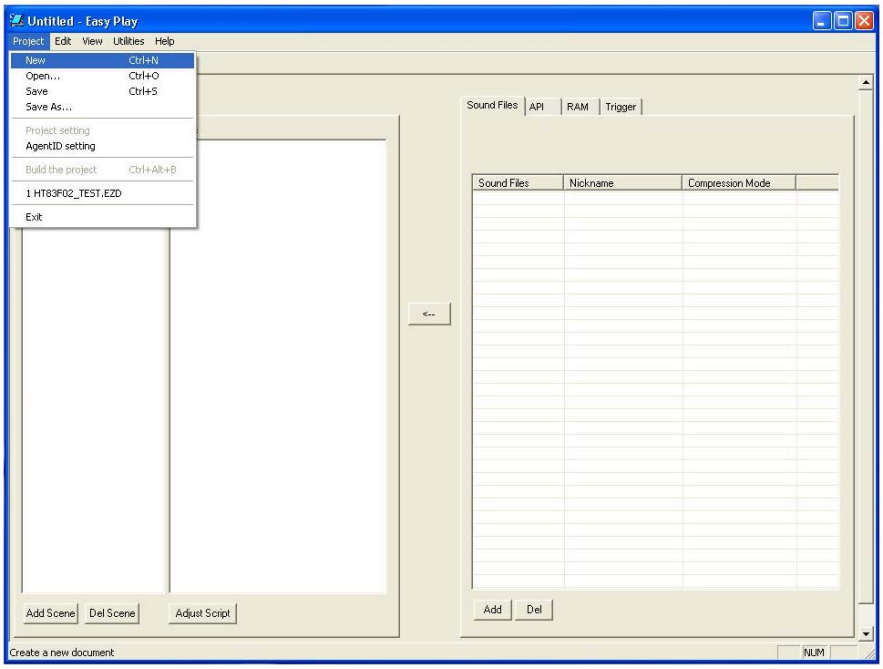
- Build a project using the Direct-8 mode in the HT83F02 according to requirements.
- The window content of the EasyPlay Scenes and Scripts is as follows:

Scenes	Scripts	Comments
Start	Selection = 0x00	(1) The parameter of the current playing sequence of the sound file. The given value is zero. (2) See EasyPlay RAM in this example.
	Max_Selection = 0x08	The parameter of the maximum playing sequence. The given value is 8.
	Zero = 0x00	The value given to Zero is 0.
	Max_Vol = 0x08	Maximum play volume parameter. The given value is 8.
	Vol = Volume	Volume parameter. The given value is the current level of the played volume.
	KeyTrigger(KeyTrigger)	(1) Initialise the key trigger "Scene". (2) See EasyPlay Trigger for reference.
	Sleep()	MCU enters the sleep mode.
Play_Current	DispatchWave(Selection)	(1) Dispatch the sound file by the Selection value. (2)See DispatchWave for reference.
	Sleep()	MCU enters the sleep mode.
Play_PREV	Jump to Selection_to_Min if Selection = Zero + 0x00	When the playing sequence parameter reaches the minimum, jump to the "Selection_to_Min" scene.
	Selection = Selection - 0x01	Or the playing sequence parameter will be minus one.
	JumpTo(Play_Current)	Jump to the "Play_Current" scene.
Play_NEXT	Selection = Selection + 0x01	Play sequence parameter plus 1.
	Jump to Selection_to_Max if Selection = Max_Selection + 0x00	When the playing sequence parameter reaches the maximum value, jump to the "Selection_to_Max" scene.
	JumpTo(Play_Current)	Or jump to the Play_Current scene.

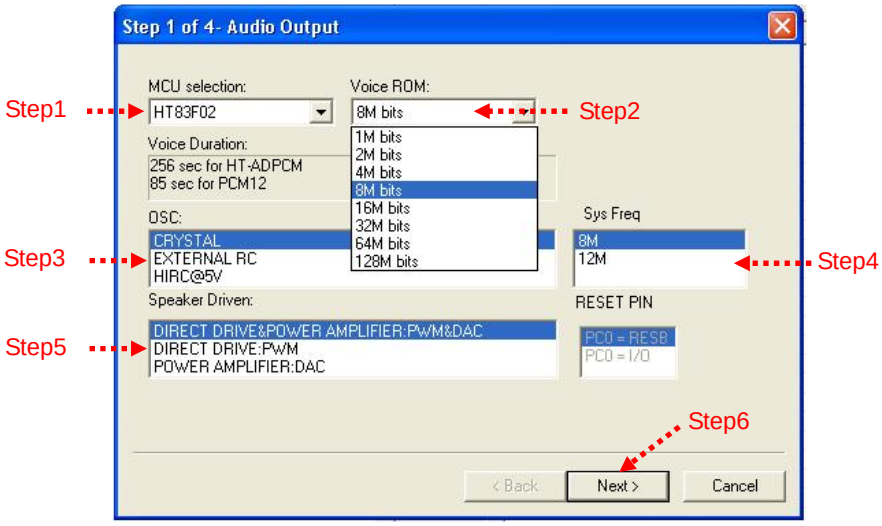
Scenes	Scripts	Comments
Selection_to_Max	Selection = Zero + 0x00	Return to the first playing sequence.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
Selection_to_Min	Selection = Max_Selection - 0x01	Return to the last playing sequence.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
Volume_Control	Dispatch(Vol)	(1) Dispatch the volume level with the Vol value. (2) See Dispatch for reference.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
	Sleep()	MCU enters the sleep mode.
Volume -	Jump to Vol_to_Min if Vol = Zero + 0x00	When the volume parameter reaches the minimum, jump to the "Vol_to_Min" scene.
	Vol = Vol - 0x01	Or the volume parameter will be minus 1.
	JumpTo(Volume_Control)	Jump to the Volume_Control scene.
Volume +	Vol = Vol + 0x01	Volume parameter plus 1.
	Jump to Vol_to_Max if Vol = Max_Vol + 0x00	When the volume parameter reaches the maximum, jump to the "Vol_to_Max" scene.
	JumpTo(Volume_Control)	Or jump to the Volume_Control scene.
Vol_to_Min	Vol = Max_Vol - 0x01	The volume level returns to the end one.
	JumpTo(Volume_Control)	Jump to the Volume_Control scene.
Vol_to_Max	Vol = Zero + 0x00	The volume level returns to the first one.
	JumpTo(Volume_Control)	Jump to the Volume_Control scene.
V0	Volume = 0x00	The current volume level is the mute status.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
V1	Volume = 0x01	The current volume level has a given value of 1.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
V2	Volume = 0x02	The current volume level has a given value of 2.
	JumpTo(Play_Current)	Jump to the Play_Current scene.
V3	Volume = 0x03	The current volume level has a given value of 3.
	JumpTo(Play_Current);	Jump to the Play_Current scene.
V4	Volume = 0x04;	The current volume level has a given value of 4.
	JumpTo(Play_Current);	Jump to the Play_Current scene.
V5	Volume = 0x05;	The current volume level has a given value of 5.
	JumpTo(Play_Current);	Jump to the Play_Current scene.
V6	Volume = 0x06;	The current volume level has a given value of 6.
	JumpTo(Play_Current);	Jump to the Play_Current scene.
V7	Volume = 0x07;	The current volume level has a given value of 7, the maximum.
	JumpTo(Play_Current);	Jump to the Play_Current scene.

Demonstration Figure Description

- Enable the EasyPlay, select "Project" from the menu and click "New" to pop up an "Audio Output" window.



- The "Audio Output" window in the EasyPlay is shown below.



In the "Audio Output" window, there are five configuration columns:

Step 1. MCU selection: Select the HT83F02.

Step 2. Voice ROM: Select the Voice ROM capacity to be 8M bits.

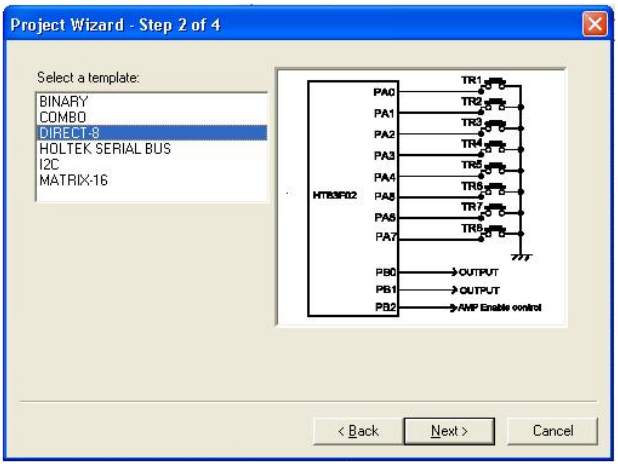
Step 3. OSC: Select the MCU oscillating mode to be HIRC@5V.

Step 4. Sys Freq: Select the MCU system operating frequency to be 12M.

Step 5. Speaker Driver: Select the speaker driving method to be PWM&DAC.

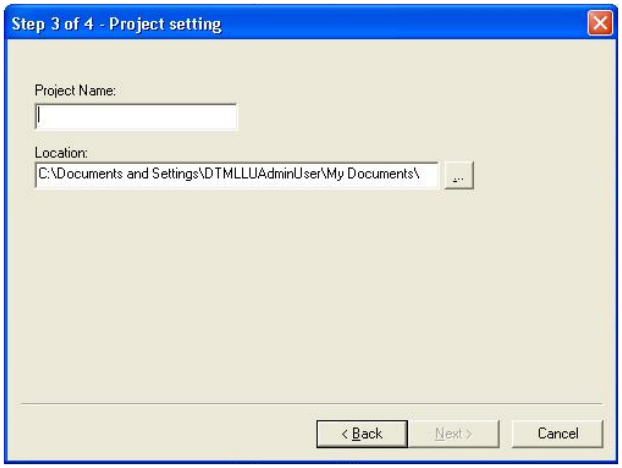
Step 6. If the user is already in the "Audio Output" window, after all the configurations are set, click the "Next" button, after which a "Project Wizard" window will pop up.

- There are six operating modes in the EasyPlay. Choose DIRECT-8 for the example here.



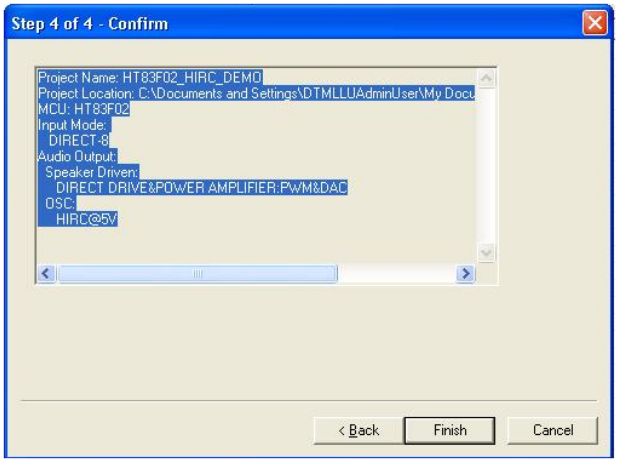
After setting up the EasyPlay operating mode, click "Next" and a "Project setting" window will pop up.

- Input the project name and setup the storage location.

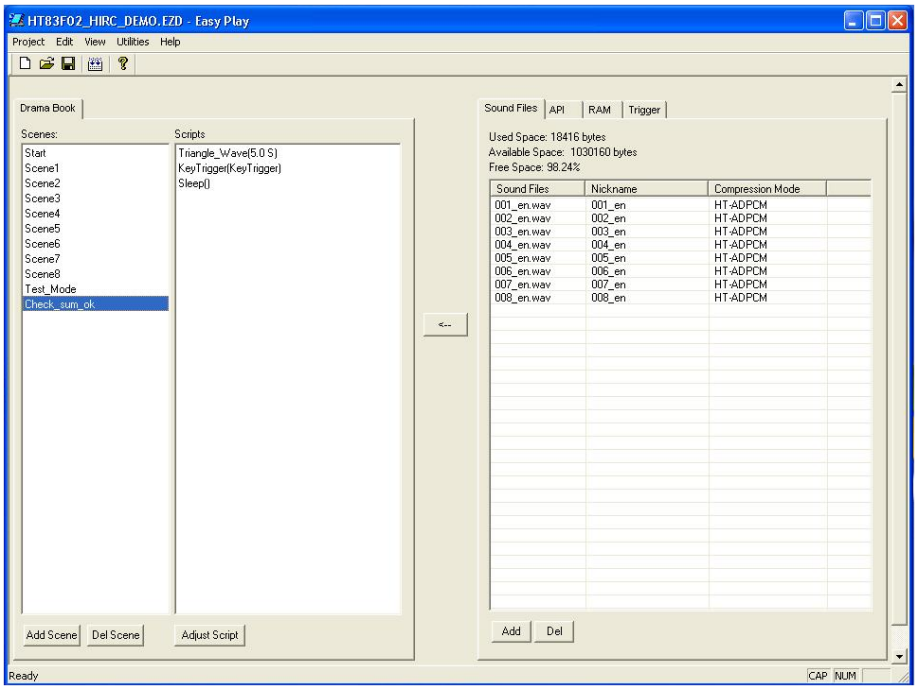


Click "Next" and a "Confirm" window will pop up.

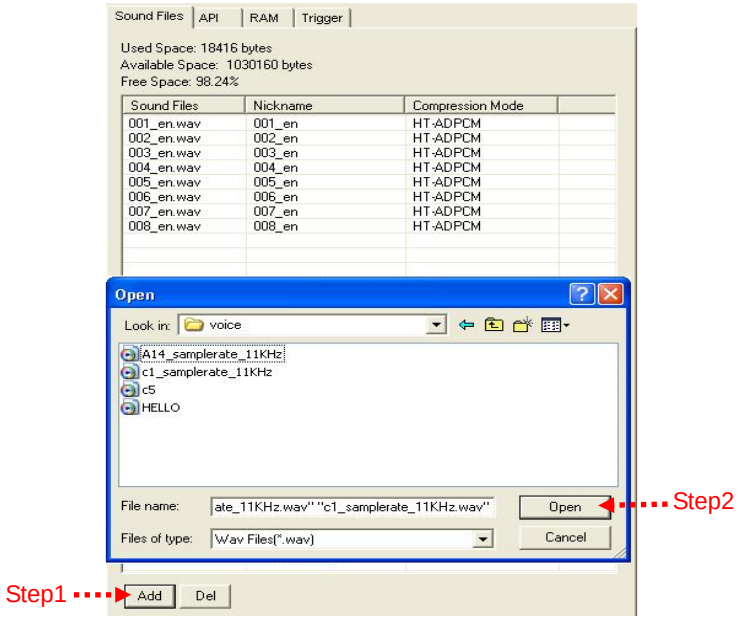
- Confirm the summarised content.



- The default basic structure setup is now complete



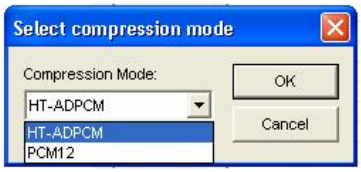
- Add a sound file.



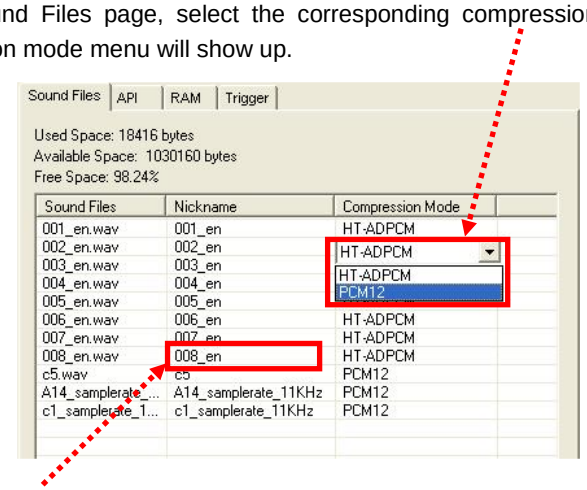
Step 1. In the Sound Files page, click "Add" and an "Open" window will pop up.

Step 2. Select a sound file, click "Open" and a Select Compression Mode window will pop up.

- Select the compression mode:
  - Following the above, select the compression mode from the Select Compression Mode window.

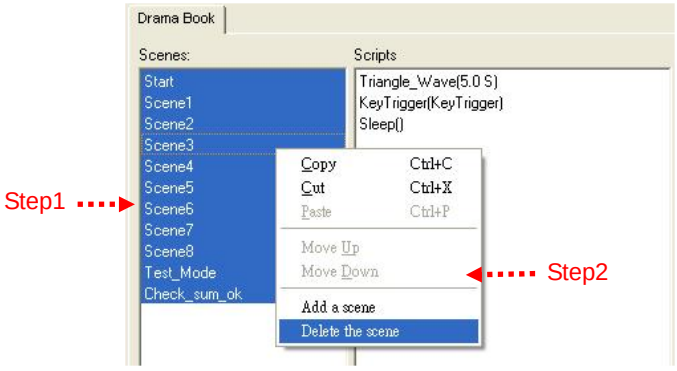


  - In the Sound Files page, select the corresponding compression mode and the compression mode menu will show up.



Note: Click Nickname column to modify the name.

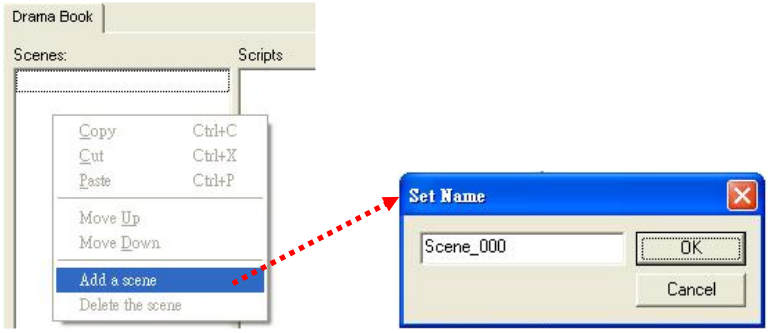
- Delete Scenes



Step 1. Select all files in the Scenes column and click the right key.

Step 2. Choose "Delete the scene" to delete the original content.

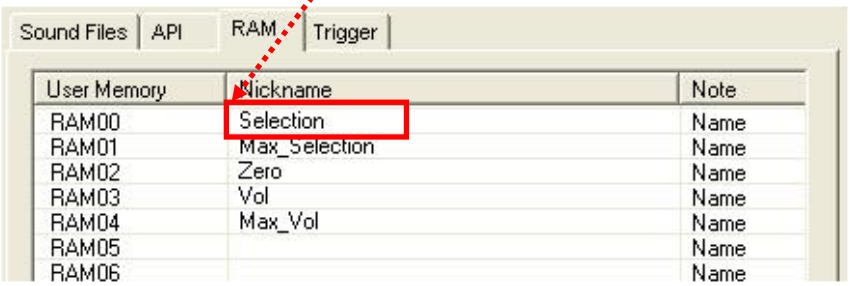
- Add Scenes and Set a Name
  - Click the right key in the Scenes dialogue, choose "Add a scene" and the Set Name dialogue will pop up for the user to set a name.



  - This shows all the Scene names in this example.

- Modify the Parameter Name

  In the RAM page, click the Nickname column to input the parameter name.



- Give values:

  Step1. Choose "Assign" from the API page.

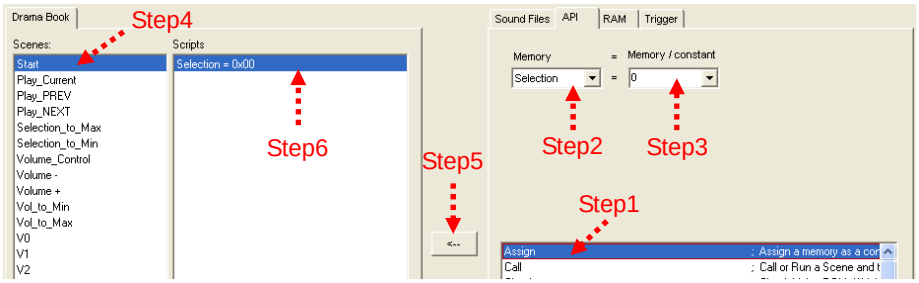
  Step2. Choose the RAM parameter from the Memory column and select a RAM parameter.

  Step3. Choose a RAM parameter or fill in a constant with a range of 0~255 in the Memory / Constant column.

  Step4. Choose the Scene to be edited from the Scenes dialogue.

  Step5. Add the API function to the Scripts.

  Step6. The API function will now be displayed in the Scripts.



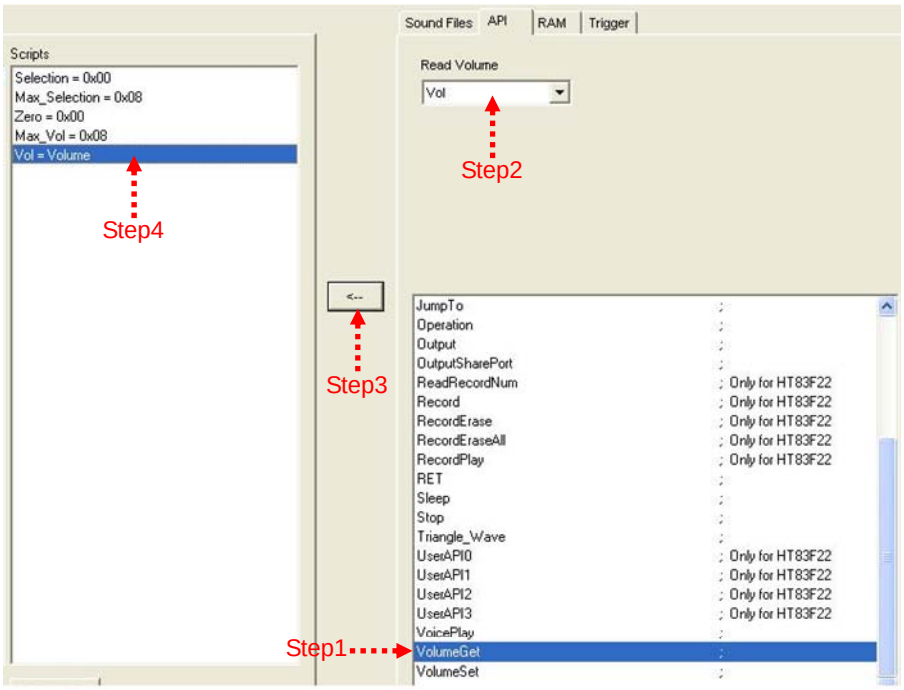
- Acquire the Sound Volume

  Step1. Choose "VolumeGet" from the API page.

  Step2. Choose a RAM parameter from the Read Volume column.

  Step3. Add the API function to the Scripts.

  Step4. The API function will now be displayed in the Scripts.



- Edit the Trigger Content

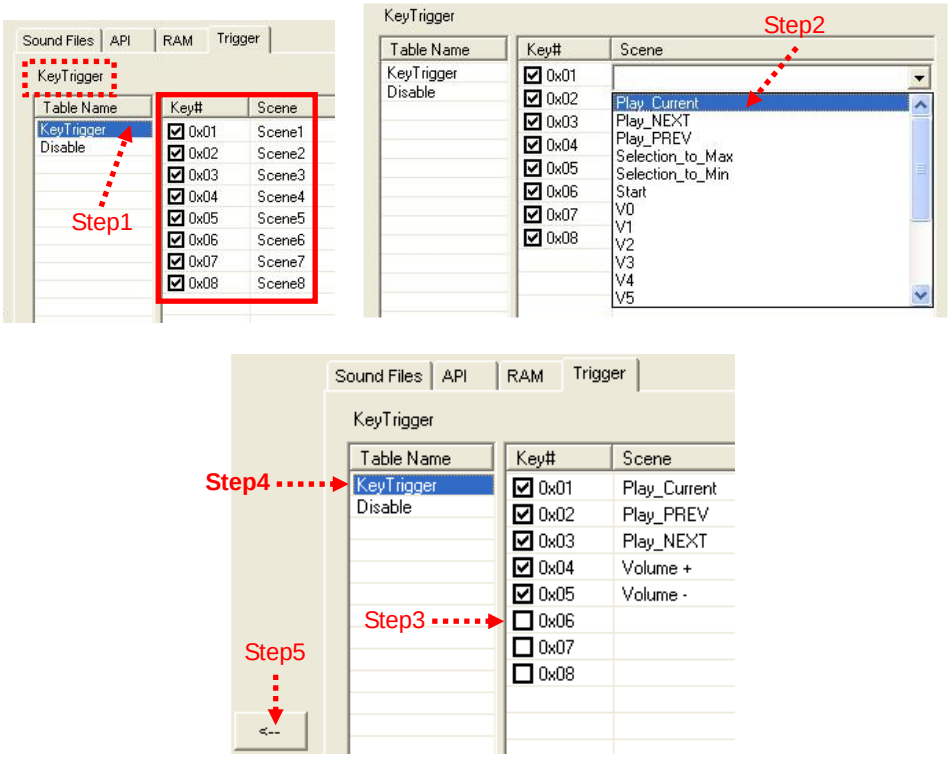
  Step1. Click the "KeyTrigger" in the Trigger page, and "KeyTrigger" will be displayed as shown the red dotted box below. The related content of the key number (key#) and the scene will be shown in the red box below.

  Step2. Click the column as the red arrow indicates for a scene selection menu.

  Step3. A box without a check mark means no function.

  Step4. Click the "KeyTrigger" column in this example.

  Step5. Add the function to the Scripts.



EasyPlay Trigger Window Content

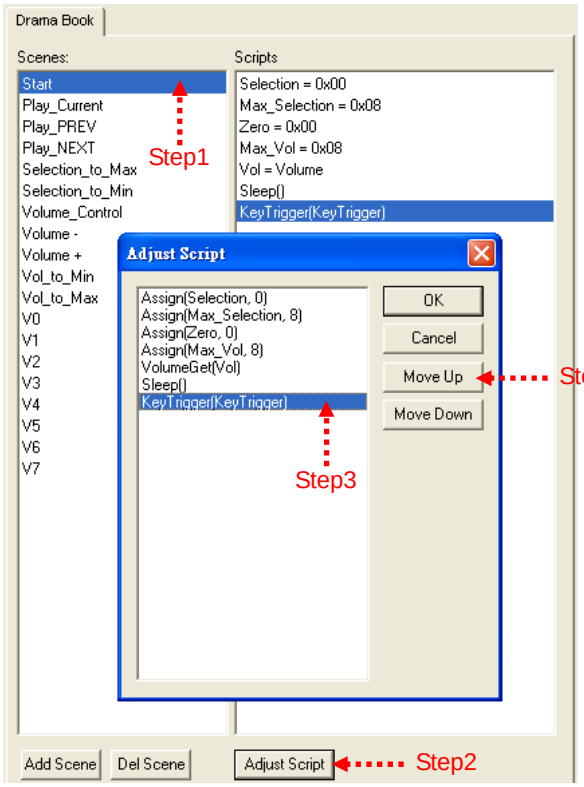
- Adjust the Scripts Sequence
  - Method I

    Step1. Click the scene to be adjusted from the Scripts column. Choose "Start" in this example.

    Step2. Click "Adjust Script" and an "Adjust Script" dialogue will pop up.

    Step3. Click the item to be adjusted.

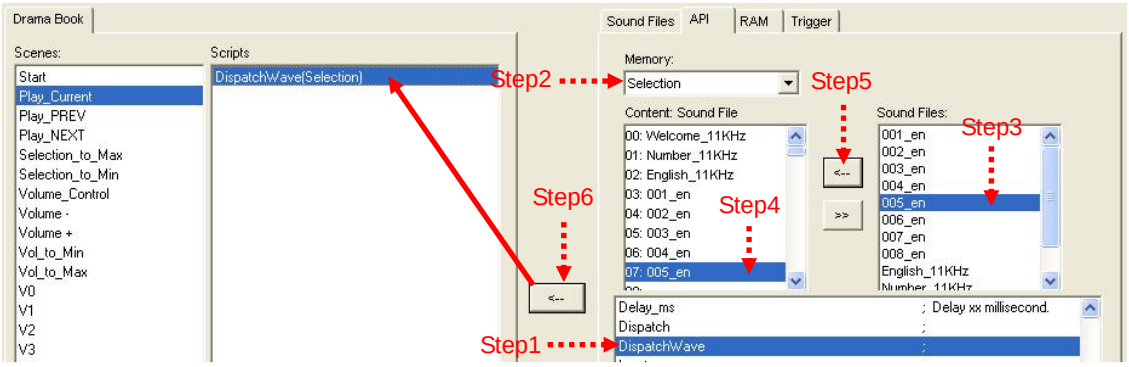
    Step4. Click "Move Up" and the item to be adjusted will shift up and then click "OK" when it reaches a proper location.



  - Method II

    Click the item to be adjusted from the Scripts column and shift it by dragging up and down.

- DispatchWave



Step1. Click "DispatchWave" in the API page to display the function column.

Step2. Choose the RAM parameter from the Memory column.

Step3. Click the required sound file.

Step4. In the Content: Sound File dialogue, choose a proper location to place the required sound file.

Step5. Add the sound file in Step 3 to the Step 4 location.

Step6. Add the API function to the Scripts.

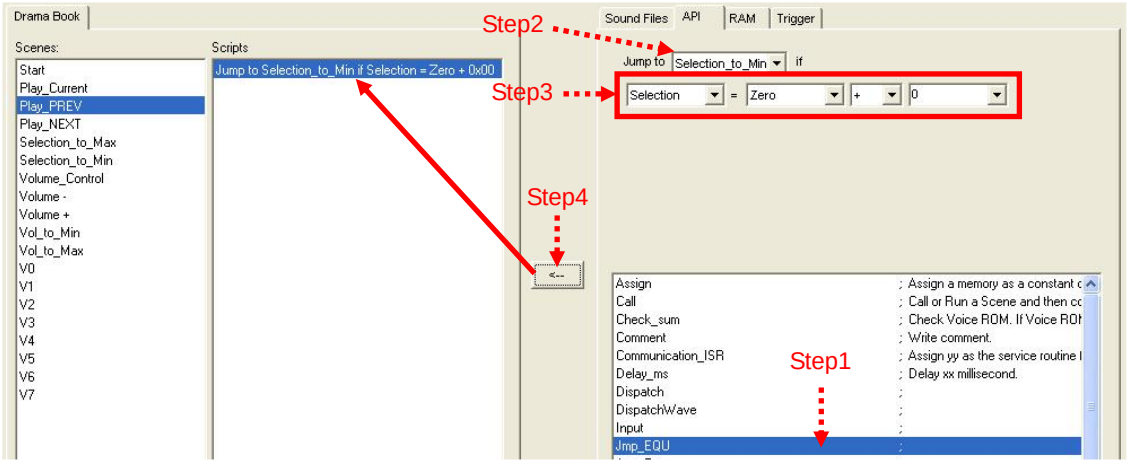
- Conditional Scene Jumping

Step1. Click "Jmp_EQU" in the API page and the function column will show up.

Step2. Click the scene to jump to.

Step3. Choose the proper parameters for the left and right side of the equal sign according to requirements.

Step4. Add the API function to the Scripts.



- Operation Application

  Step1. Click "Operation" in the API page to show the function column.

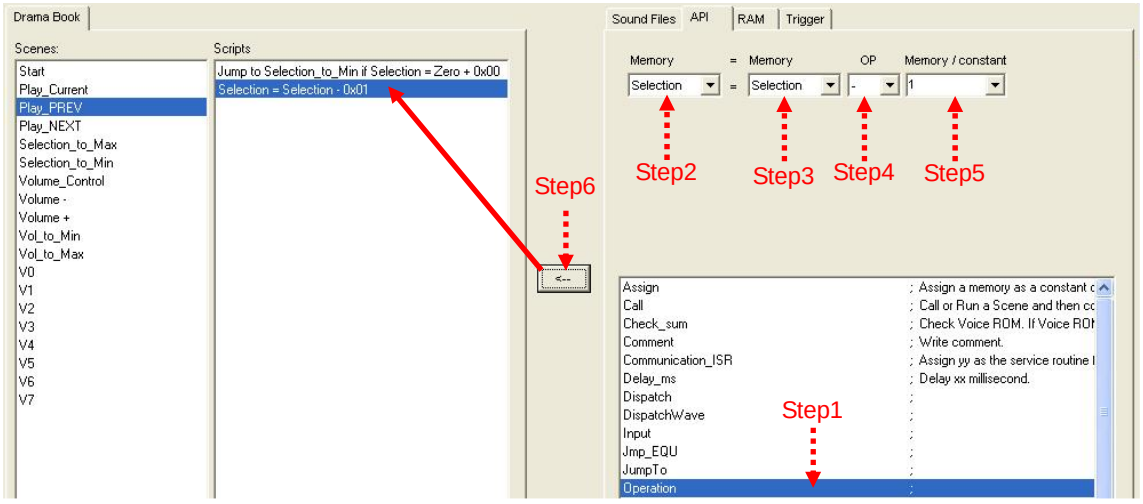
  Step2. Place the operation result at the right side of the equal sign to the selected RAM parameter.

  Step3. Choose the RAM parameter according to requirements.

  Step4. Choose the operation symbol according to requirements.

  Step5. Choose the RAM parameter or add a constant with a range of 0~255 according to requirements.

  Step6. Add the API function to the Scripts.

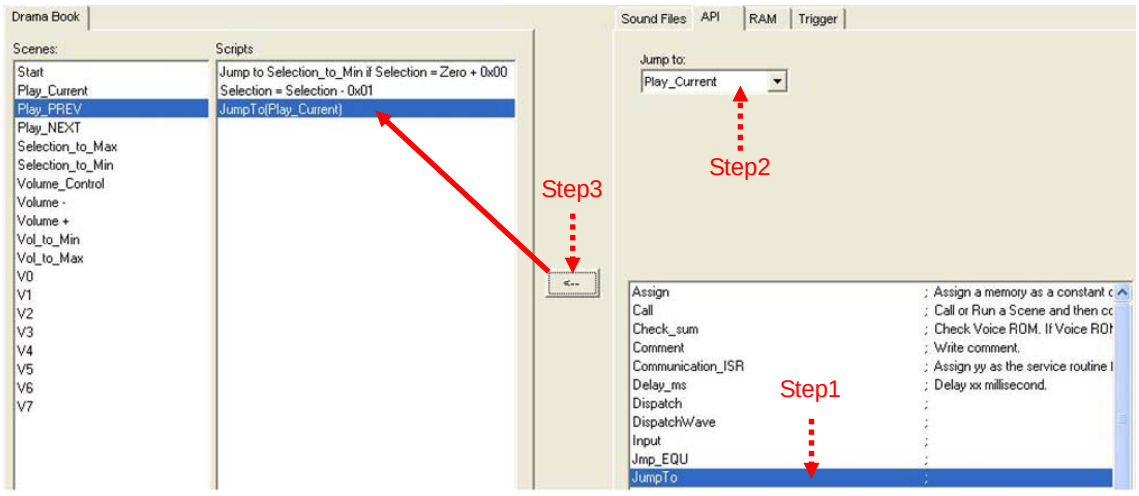


- No Conditional Scene Jumping

  Step1. Click "JumpTo" in the API page to display the function column.

  Step2. Choose the scene to jump to.

  Step3. Add the API function to the Scripts.



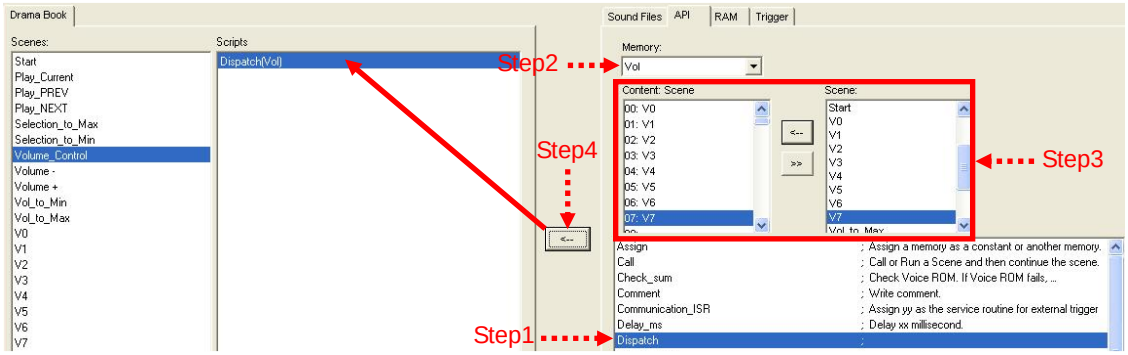
- Dispatch Content

  Step1. Click "Dispatch" in the API page to display the function column.

  Step2. Choose the RAM parameter from the Memory column.

  Step5. According to requirements add the "Scene:" items to the "Content: Scene" column.

  Step6. Add the API function to the Scripts.

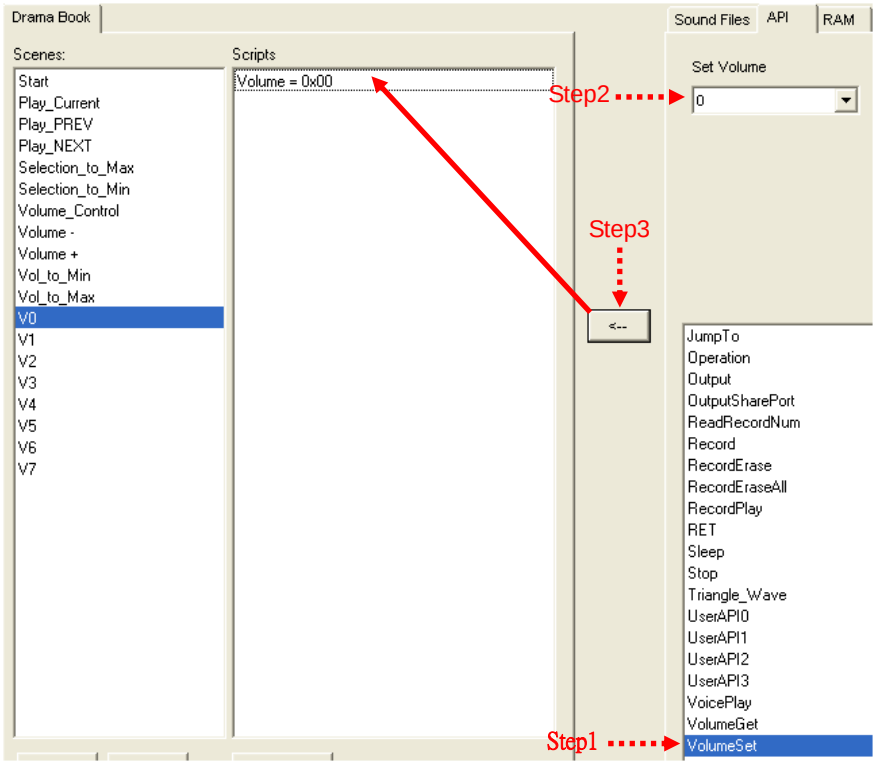


- Volume Setting

  Step1. Click "VolumeSet" in the API page to display the function column.

  Step2. Choose a value 0-7 from the Set Volume column.

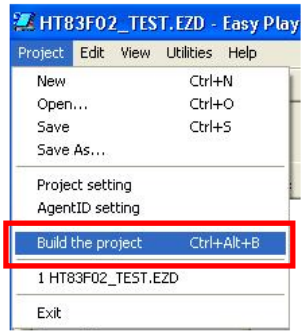
  Step3. Add the API function to the Scripts.



Programming and Demo Environments

Create .PND file

- Method I
  Choose the following function according to the location

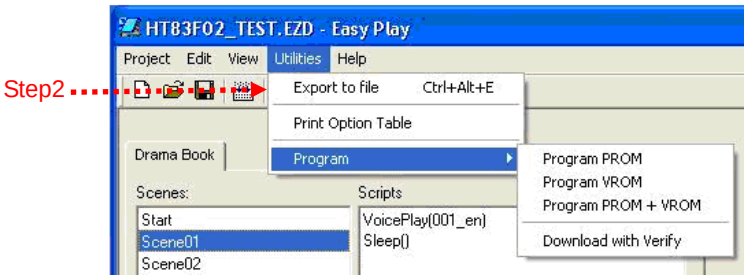


- Method II
  Click the icon in the red circle

Programming

Step1. Based on Fig.1, create the hardware platform in advance.

Step2. Choose the programming method according to requirements. See details in the Project Design Chapter.



Programming and Demo Environments

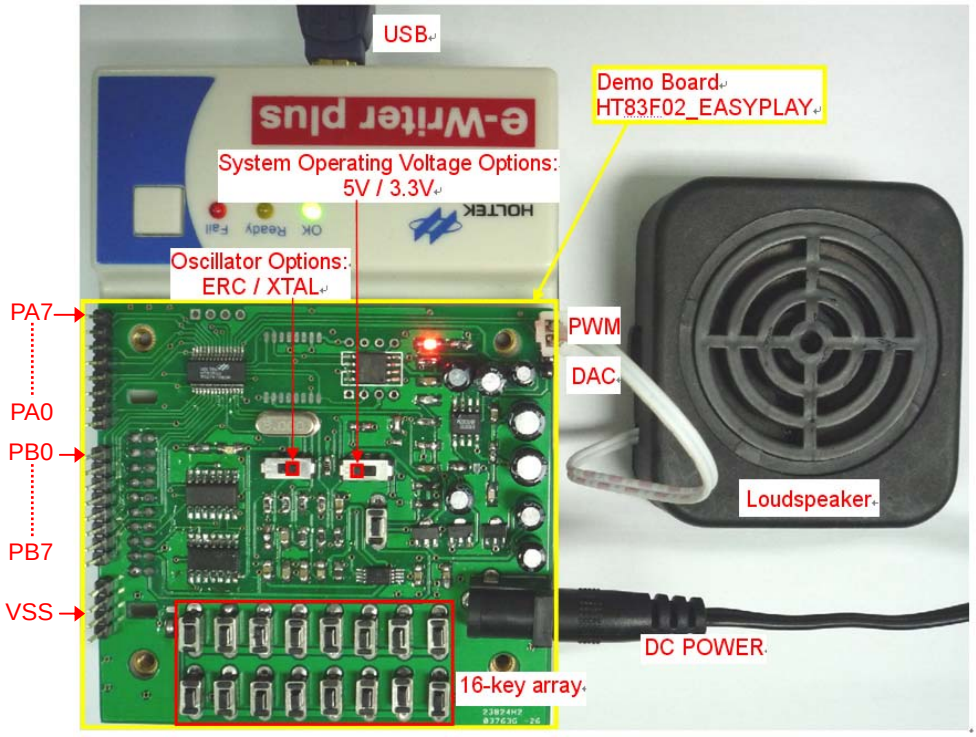


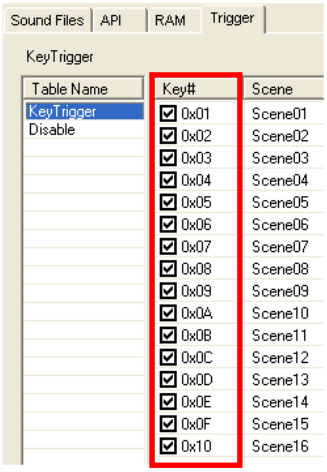
Fig.1

Note: 1. The DC POWER is a DC9V~12V power supply which will automatically switch to the USB power during programming. The external power is only for demonstration purposes and is not used for the programming process.

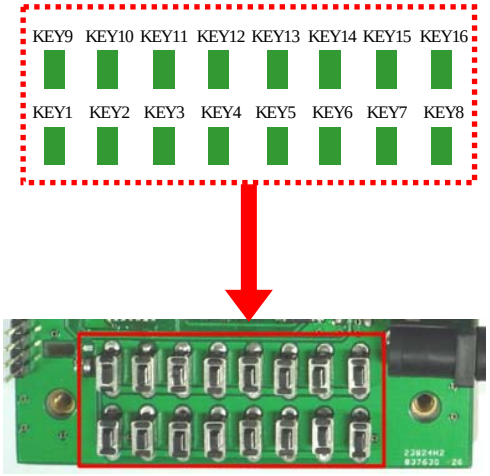
2. After the USB programming has finished, no external power is needed for demonstration.

3. Key Diagram

A. Software Platform Key Number (Key#)



B. Hardware Platform Key Number



Key and Transmission Applications

Key Requirements

- The example uses the HT83F02 MCU.
- Six keys:
  K1 -- Play a single song once
  K2 -- Continuously play 8 songs once
  K3 -- Non-stop playing
  K4 -- Stop playing
  K5 -- Increase volume
  K6 -- Decrease volume
- I^2C communication data can be received from an external master device and according to the received data can be related to the key function.

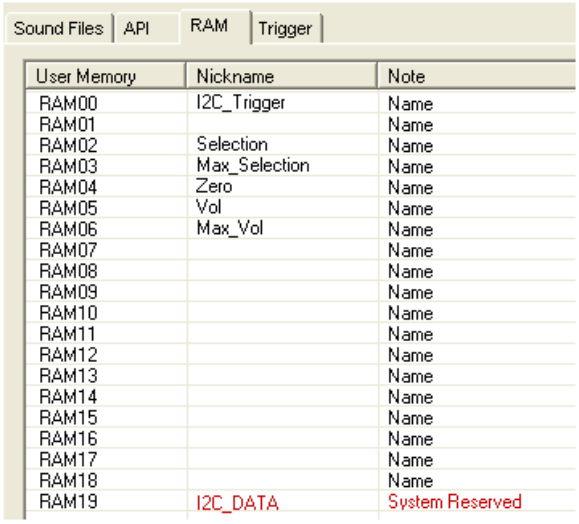
Functional Description

- Build a project using the COMBO mode according to the selected HT83F02 requirements.
- The window content of the EasyPlay Scenes and Scripts is as follows:

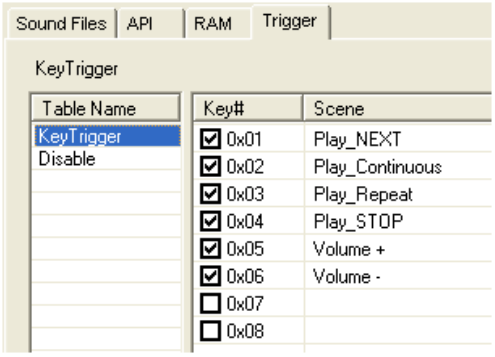
Scenes	Scripts	Comments
Start	KeyTrigger(Disable);	Key trigger function is disabled.
	Selection = 0x00;	(1) Sound file play sequence parameter. The given value is zero. (2) See EasyPlay RAM for this example.
	Max_Selection = 0x08	Maximum playing sequence parameter. The given value is 8.
	Zero = 0x00;	The value given to Zero is 0.
	Max_Vol = 0x08;	Maximum playing volume parameter. The given value is 8.
	Vol = Volume;	Volume parameter. The given value is the current level of the volume played.
	Communication_ISR(I2C_ISR);	Initialize the data communication "Scene".
	KeyTrigger(KeyTrigger);	(1) Initialise the key trigger "Scene". (2) See EasyPlay Trigger for reference.
	Sleep();	MCU enters the sleep mode.
Play_Current	DispatchWave(Selection);	(1) Dispatch the sound file using the Selection value. (2) See DispatchWave for reference.
	Sleep();	MCU enters the sleep mode.
Play_NEXT	Selection = Selection + 0x01;	Play sequence parameter plus 1.
	Jump to Selection_to_Max if Selection = Max_Selection + 0x00;	When the playing sequence parameter reaches the maximum value, jump to the "Selection_to_Max" scene.
	JumpTo(Play_Current);	Or jump to the Play_Current scene.
Selection_to_Max	Selection = Zero + 0x00;	Return to the first play sequence0.
	JumpTo(Play_Current);	Jump to the Play_Current scene.
Play_Continuous	VoicePlay(001_en);	Play the sound file "001_en"
	VoicePlay(002_en);	Play the sound file "002_en"
	VoicePlay(003_en);	Play the sound file "003_en"
	VoicePlay(004_en);	Play the sound file "004_en"
	VoicePlay(005_en);	Play the sound file "005_en"
	VoicePlay(006_en);	Play the sound file "006_en"
	VoicePlay(007_en);	Play the sound file "007_en"
	VoicePlay(008_en);	Play the sound file "008_en"
	Sleep();	MCU enters the sleep mode.
Play_Repeat	VoicePlay(001_en);	Play the sound file "001_en"
	VoicePlay(002_en);	Play the sound file "002_en"
	VoicePlay(003_en);	Play the sound file "003_en"
	VoicePlay(004_en);	Play the sound file "004_en"
	VoicePlay(005_en);	Play the sound file "005_en"

Scenes	Scripts	Comments
	VoicePlay(006_en);	Play the sound file "006_en"
	VoicePlay(007_en);	Play the sound file "007_en"
	VoicePlay(008_en);	Play the sound file "008_en"
	JumpTo(Play_Repeat);	Jump to the Play_Repeat scene
Play_STOP	Stop(CH0);	Stop playing.
	Sleep();	MCU enters the sleep mode.
Volume_Control	Dispatch(Vol);	(1) Dispatch the volume level with the Vol value. (2) See Volume Setting for reference.
	Sleep();	MCU enters the sleep mode.
Volume -	Jump to Vol_to_Min if Vol = Zero + 0x00;	When the volume parameter reaches the minimum, jump to the "Vol_to_Min" scene.
	Vol = Vol - 0x01;	Or the volume parameter will reduce by 1.
	JumpTo(Volume_Control);	Jump to the Volume_Control scene.
Volume +	Vol = Vol + 0x01;	Volume parameter plus 1.
	Jump to Vol_to_Max if Vol = Max_Vol + 0x00;	When the volume parameter reaches the maximum, jump to the "Vol_to_Max" scene.
	JumpTo(Volume_Control);	Or jump to the Volume_Control scene.
Vol_to_Min	Vol = Max_Vol - 0x01;	The volume level returns to the end one.
	JumpTo(Volume_Control);	Jump to the Volume_Control scene.
Vol_to_Max	Vol = Zero + 0x00;	The volume level returns to the first one.
	JumpTo(Volume_Control);	Jump to the Volume_Control scene.
V0	Volume = 0x00	The current volume level is the mute status.
	Sleep();	MCU enters the sleep mode.
V1	Volume = 0x01;	The current volume level has a given value of 1.
	Sleep();	MCU enters the sleep mode.
V2	Volume = 0x02;	The current volume level has a given value of 2.
	Sleep();	MCU enters the sleep mode.
V3	Volume = 0x03;	The current volume level has a given value of 3.
	Sleep();	MCU enters the sleep mode.
V4	Volume = 0x04;	The current volume level has a given value of 4.
	Sleep();	MCU enters the sleep mode.
V5	Volume = 0x05;	The current volume level has a given value of 5.
	Sleep();	MCU enters the sleep mode.
V6	Volume = 0x06;	The current volume level has a given value of 6
	Sleep();	MCU enters the sleep mode.
V7	Volume = 0x07;	The current volume level has a given value of 7, the maximum.
	Sleep();	MCU enters the sleep mode.
I2C_ISR	I2C_Trigger = I2C_DATA AND 0x07;	Transmit the lowest 3 bits of the I2C_DATA to I2C_Trigger.
	Dispatch(I2C_Trigger);	See I²C Data Relation for reference.
	Sleep();	MCU enters the sleep mode.
Test_Mode	Check_sum();	Accumulate and calibrate the VROM space.
	RAM1 = 0x00;	The value given to RAM1=0.
	Jump to Check_sum_ok if RAM18 = RAM1 + 0x00;	If the accumulation and calibration of VROM is correct, then RAM18=0 or else RAM18=1. Jump to Check_sum_ok if it is correct.
	Triangle_Wave(1.0 S);	Incorrect, the AUDIO port outputs a 1S triangle wave.
	Delay_ms(1000);	Pause for 1S.
	Triangle_Wave(1.0 S);	Re-output 1S triangle wave.
	KeyTrigger(KeyTrigger);	Initialise the key trigger "Scene".
	Sleep();	MCU enters the sleep mode.
Check_sum_ok	Triangle_Wave(5.0 S);	If the accumulation and calibration of VROM is correct, the AUDIO port outputs a 5S triangle wave.
	KeyTrigger(KeyTrigger);	Initialise the key trigger "Scene".
	Sleep();	MCU enters the sleep mode.

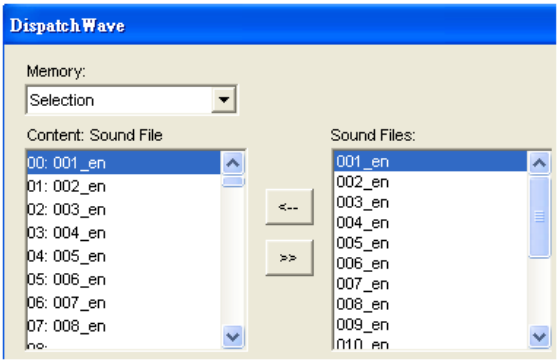
- EasyPlay RAM Window Content



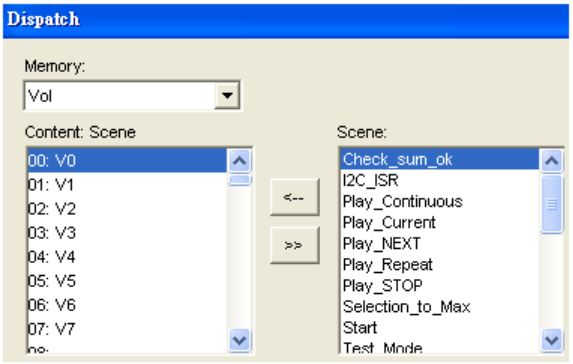
- EasyPlay Trigger Window Content



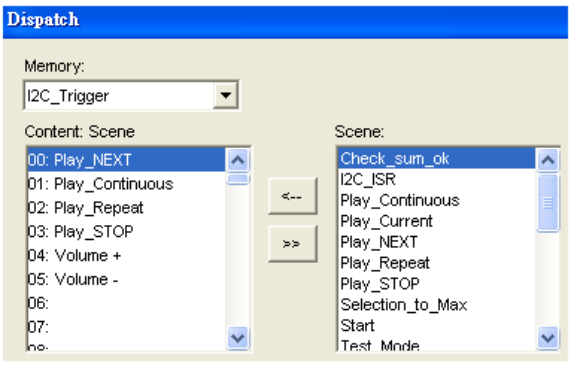
- DispatchWave Window Content



- Volume Setting Window Content



- I^2C Communication Data Relation Window Content



Recording and Audio Function Applications

Function Requirements

Five keys:

- K1 -- Erase all recordings
- K2 -- Add one recording
- K3 -- Play files in upward cycle
- K4 -- Play files in downward cycle
- K5 -- Stop recording/playing

Functional Description

- Choosing the Direct-6 mode in the HT83F22 to start a project according to requirements.
- The windows of the EasyPlay Scenes and Scripts are as follows:

Scenes	Scripts	Comments
Start	KeyTrigger(KeyTrigger);	Initialise the key trigger "Scene".
	Sleep();	MCU enters the sleep mode.
EraseAllRecord	RecordEraseAll();	Erase all recordings.
	Sleep();	MCU enters the sleep mode.
AddOneRecord	RecordIndex = ReadRecordNum;	Read the present recorded item to the recording sequence.
	Record(RecordIndex);	Record a sound.
	Sleep();	Recording completed automatically, the MCU enters the sleep mode.
PlayRecord_Up	RecordNum_max = ReadRecordNum;	Read the present recording number to the RecordNum_max.
	Jump to STOP if ZERO = RecordNum_max + 0x00;	If no recording, no file is played.
	Call(Up_PlayRecordNum);	Call the subroutine that adjusts the record/playing sequence parameter.
	RecordPlay(PlayRecordNum);	Play the recording file according to the recording sequence parameter.
	Sleep();	MCU enters the sleep mode after the recording file is played.
PlayRecord_Down	RecordNum_max = ReadRecordNum;	Read the current recording number to the RecordNum_max.
	Jump to STOP if ZERO = RecordNum_max + 0x00;	If no recording is done, no file will be played.
	Call(Down_PlayRecordNum);	Call the subroutine that adjusts the record/playing sequence parameter.
	RecordPlay(PlayRecordNum);	Play the file according to the record/playing sequence parameter.
	Sleep();	MCU enters the sleep mode after the recording file is played.
STOP	Sleep();	Stop recording or playing and enter the sleep mode.
Up_PlayRecordNum	Jump to Up_max if RecordNum_max=PlayRecordNum+0x01;	If the recording/playing sequence parameter reaches the maximum, jump to the "Up_MAX" scene.
	PlayRecordNum = PlayRecordNum+0x01;	Or the record/playing sequence parameter increases by 1.
	RET();	Return from subroutine.
Up_MAX	PlayRecordNum = 0x00;	The record/playing sequence returns to the first one.
	RET();	Return from subroutine.
Down_PlayRecordNum	Jump to Down_MIN if ZERO = PlayRecordNum + 0x00;	If the recording/playing sequence parameter reaches the minimum, jump to the "Down_MIN" scene.
	PlayRecordNum = PlayRecordNum – 0x01;	Or the recording/playing sequence parameter reduces by 1.
	RET();	Return from subroutine.
Down_MIN	PlayRecordNum = ReadRecordNum;	Read the current recording number to the parameter of the record/playing sequence.
	PlayRecordNum= PlayRecordNum – 0x01;	The record/playing sequence returns to the end.
	RET();	Return from subroutine.

- The EasyPlay RAM and Trigger window content is as follows.

Sound Files | API | RAM | Trigger

User Memory	Nickname	Note
RAM00	RecordIndex	Name
RAM01	PlayRecordNum	Name
RAM02	RecordNum_max	Name
RAM03	ZERO	Name
RAM04		Name
RAM05		Name
RAM06		Name
RAM07		Name
RAM08		Name
RAM09		Name
RAM10		Name
RAM11		Name
RAM12		Name
RAM13		Name
RAM14		Name
RAM15		Name
RAM16		Name
RAM17		Name
RAM18		Name
RAM19		Name

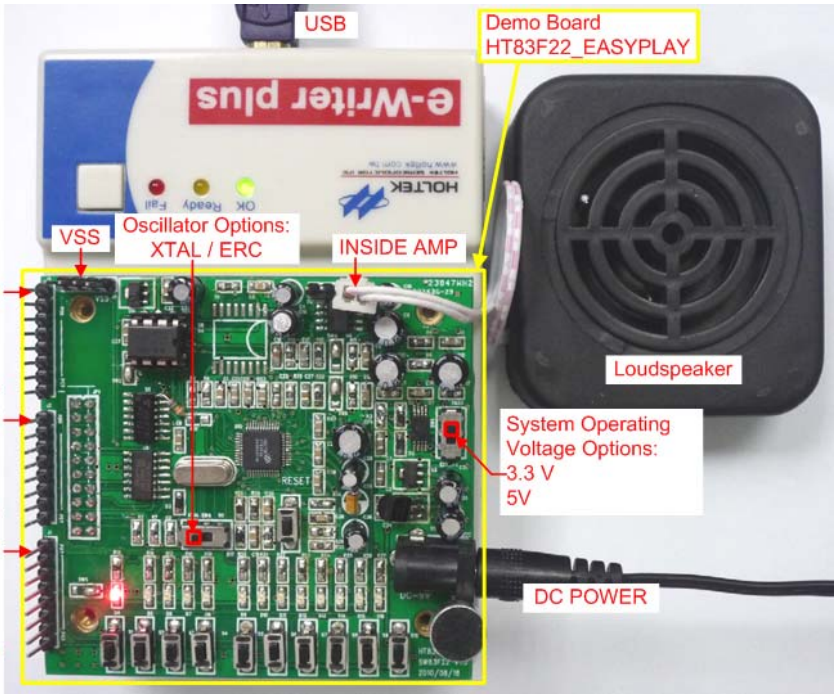
Sound Files | API | RAM | Trigger

KeyTrigger

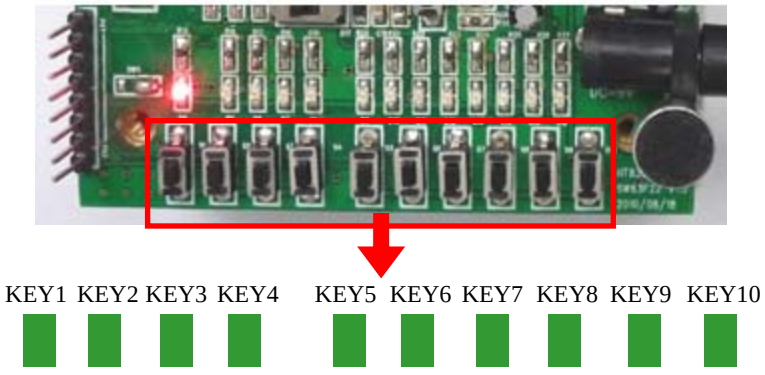
Table Name	Key#	Scene
KeyTrigger	☑ 0x01	EraseAllRecord
	☑ 0x02	AddOneRecord
	☑ 0x03	PlayRecord_Up
	☑ 0x04	PlayRecord_Down
	☑ 0x05	STOP
	☐ 0x06	

Programming and Demo Environments

- Programming and demo environment diagrams



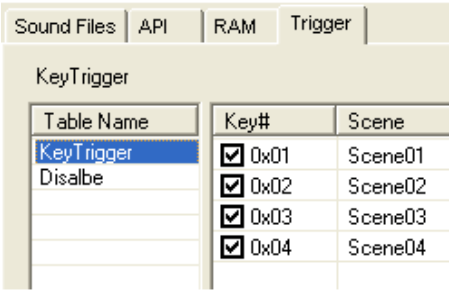
- Demo Board HT83F22_EASYPLAY Key Placement
  - Hardware Platform Key Number



– Software Platform Key Number (Key#)

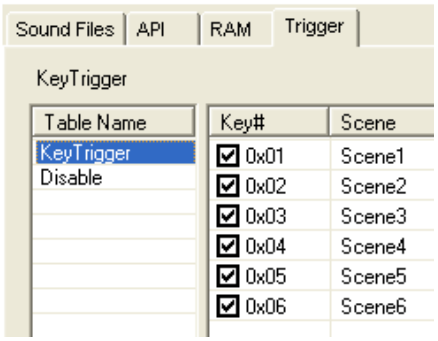
a) Combo Operating Mode

The key numbers related to the hardware are KEY5-KEY8.



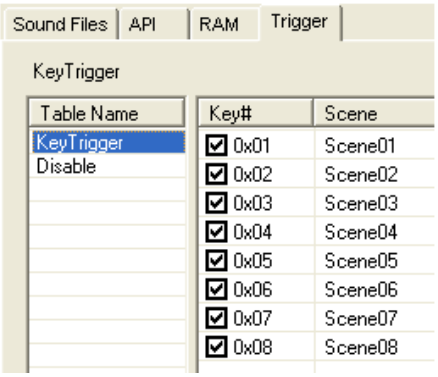
b) Direct-6 Operating Mode

The key numbers related to the software are KEY5-KEY10.



c) Matrix-8 Operating Mode

The key numbers related to the hardware are KEY1-KEY8.



Holtek Semiconductor Inc. (Headquarters)
No.3, Creation Rd. II Science Park. Hsinchu, Taiwan
Tel: 886-3-563-1999
FAX: 886-3-563-1189
http://www.holtek.com

Holtek Semiconductor Inc. (Taipei Sales Offeice)
4F-2, No. 3-2, YuanQu St., Nankang Software Park, Taipei 115, Taiwan
Tel: 886-2-2655-7070
FAX: 886-2-2655-7373
FAX: 886-2-2655-7383 (International sales hotline)

Holtek Semiconductor Inc. (Shenzhen Sales Offeice)
5F, Unit A, Productivity Building, No.5 Gaoxin M 2nd Road, Nanshan District, Shenzhen, China 518057
Tel: 86-755-8616-9908
FAX: 86-755-8616-9722

Holmate Semiconductor(USA), Inc. (North America Sales Office)
46712 Fremont Blvd., Fremont, CA 94538
Tel: 1-510-252-9880
FAX: 1-510-252-9885
Http:// www.holmate.com

Copyright © 2011 by HOLTEK SEMICONDUCTOR INC.

The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek's products are not authorized for use as critical components in life support devices or systems. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at http://www.holtek.com